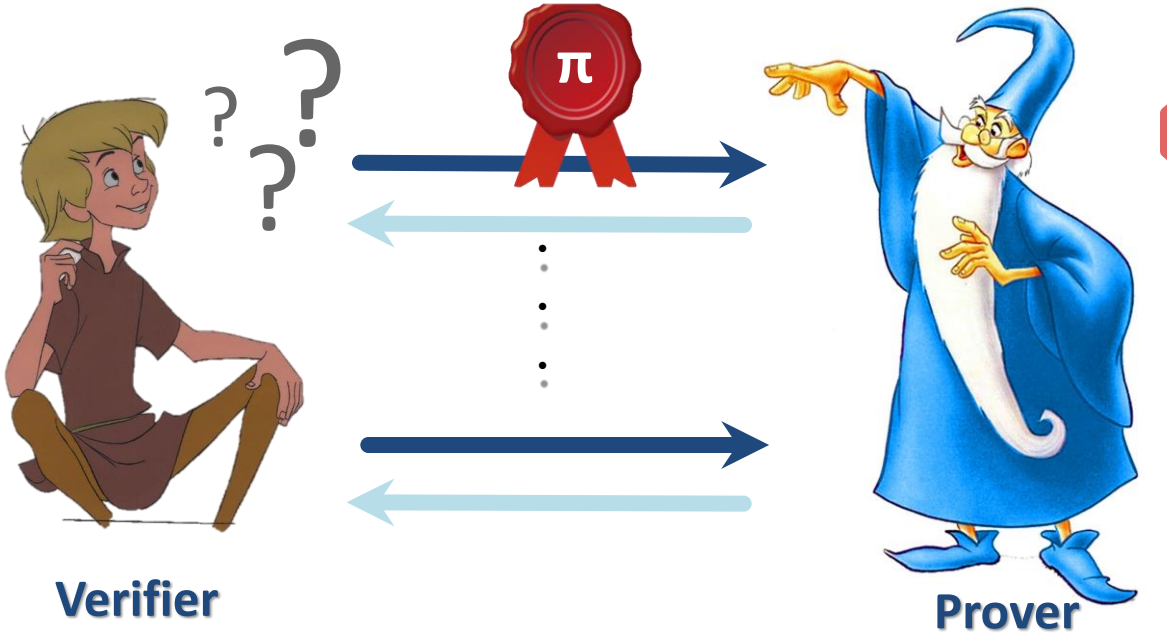
Doubly Efficient Interactive Proofs over Infinite and Non-Commutative Rings

Eduardo Soria-Vazquez

Crypto
Research Center

Doubly-Efficient Interactive proofs (a.k.a. “GKR-style”)

[GKR08] Goldwasser, S., Kalai, Y. T., & Rothblum, G. N. “Delegating computation: interactive proofs for muggles”. STOC’08.



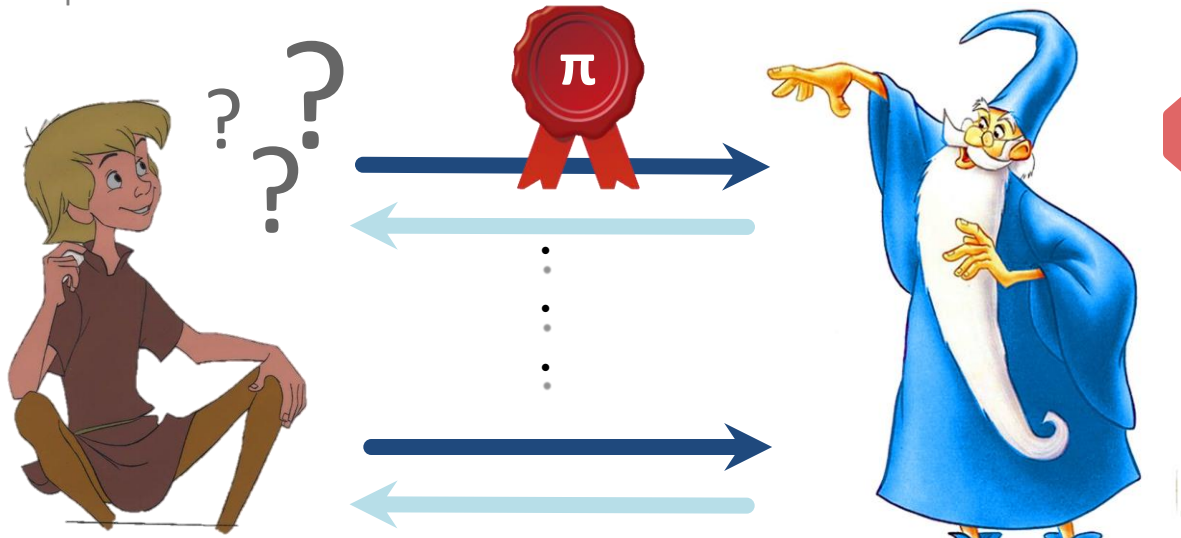
Soundness: False proofs should be rejected.

Correctness: Correct proofs should be accepted.

Doubly-Efficient Interactive proofs (a.k.a. “GKR-style”)

[GKR08] Goldwasser, S., Kalai, Y. T., & Rothblum, G. N. “Delegating computation: interactive proofs for muggles”. STOC’08.

[XZZPS19] Xie, T., Zhang, J., Zhang, Y., Papamanthou, C., & Song, D. “Libra: Succinct zero-knowledge proofs with optimal prover computation.” CRYPTO’19.



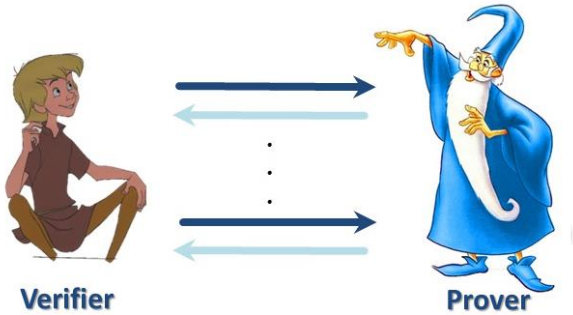
Verifier
(quasi-linear in input size, log in circuit size)

Prover
(Linear time on circuit size)

Soundness:
False proofs should be rejected.

Correctness:
Correct proofs should be accepted.

Verifiable computation

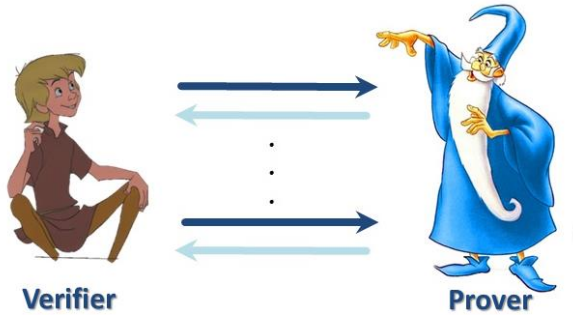


Input: Circuit C, circuit inputs x
Output: y, proof that $C(x) = y$

Verifiable computation (of complex programs, in practice)

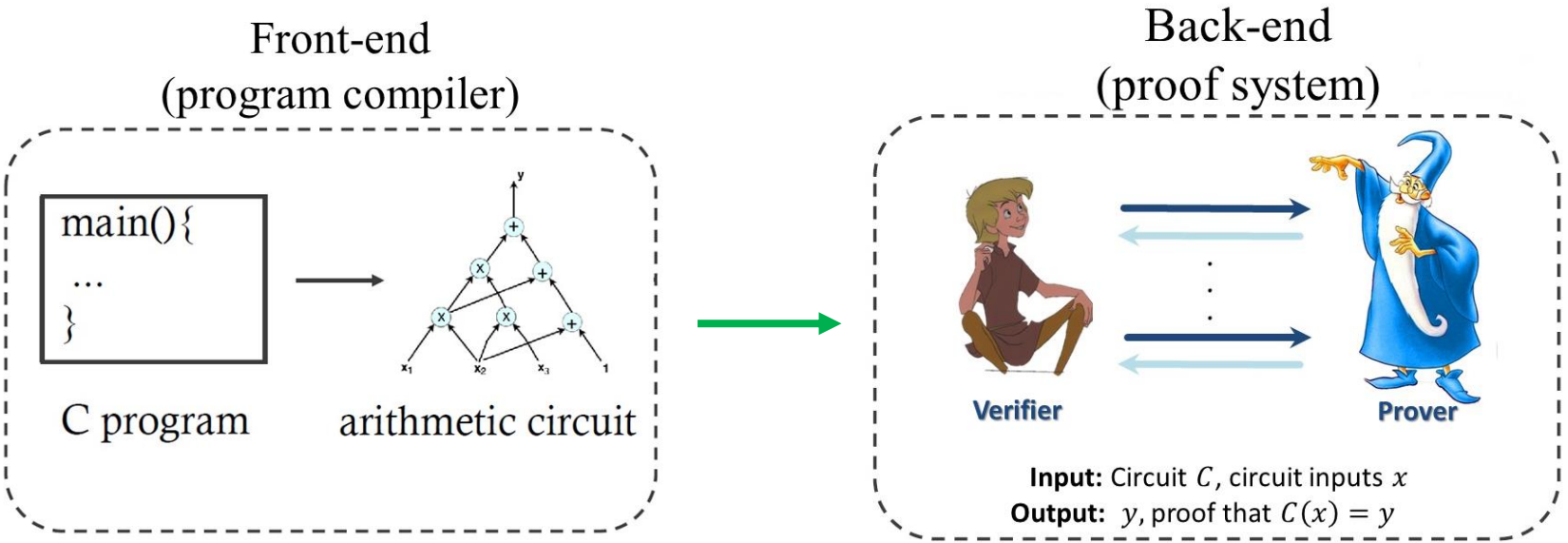
```
main(){
 ...
}
```

C program

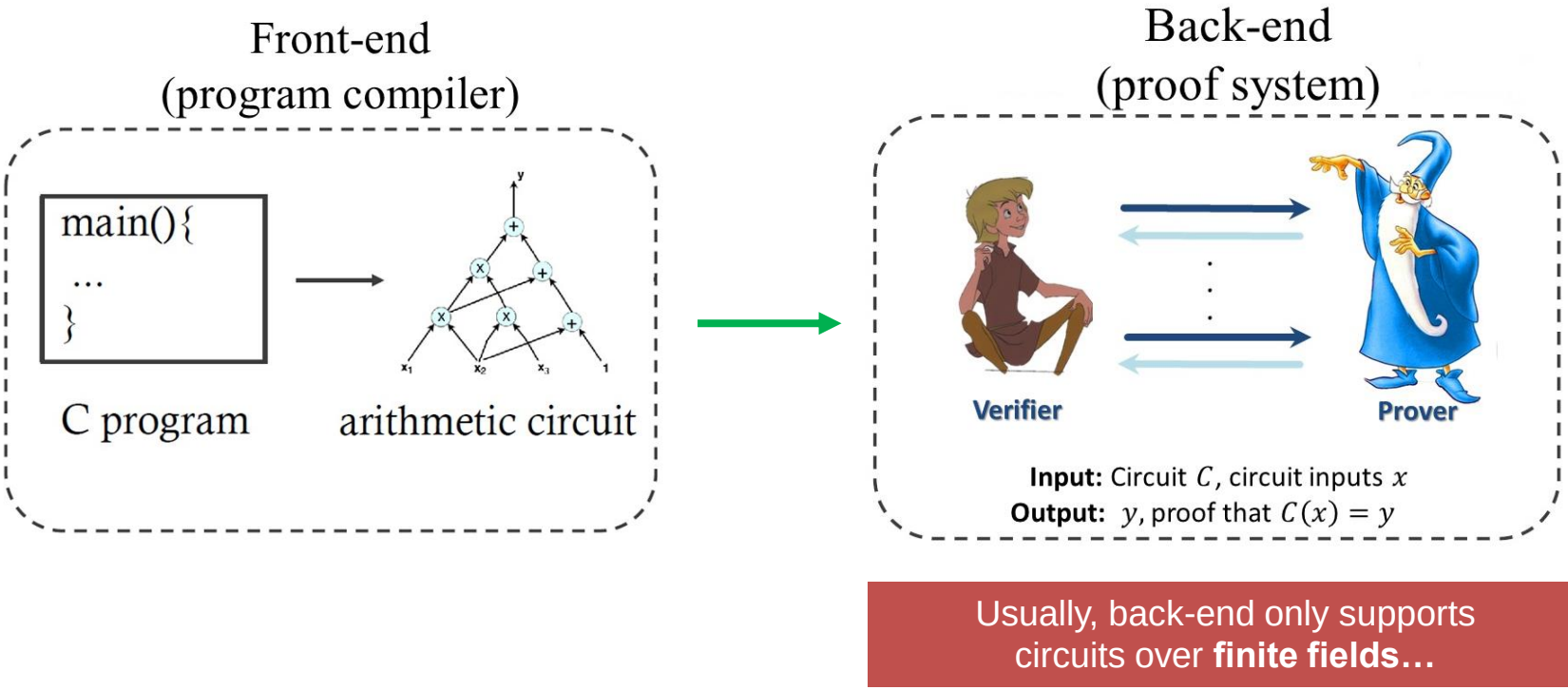


Input: Circuit C, circuit inputs x
Output: y, proof that $C(x) = y$

Verifiable computation (of complex programs, in practice)



Verifiable computation (of complex programs, in practice)



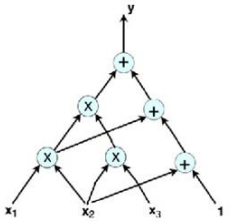
Usually, back-end only supports circuits over **finite fields…**

Verifiable computation (of complex programs, in practice)

Front-end
(program compiler)

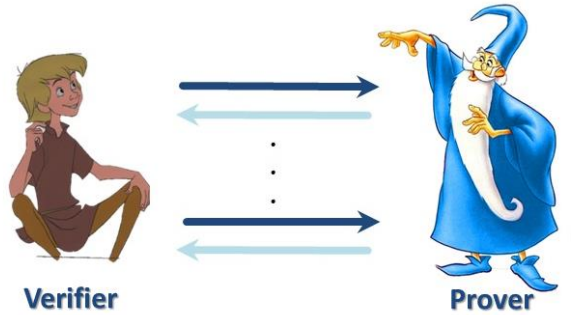
```
main(){
 ...
}
```

C program



arithmetic circuit

Back-end
(proof system)



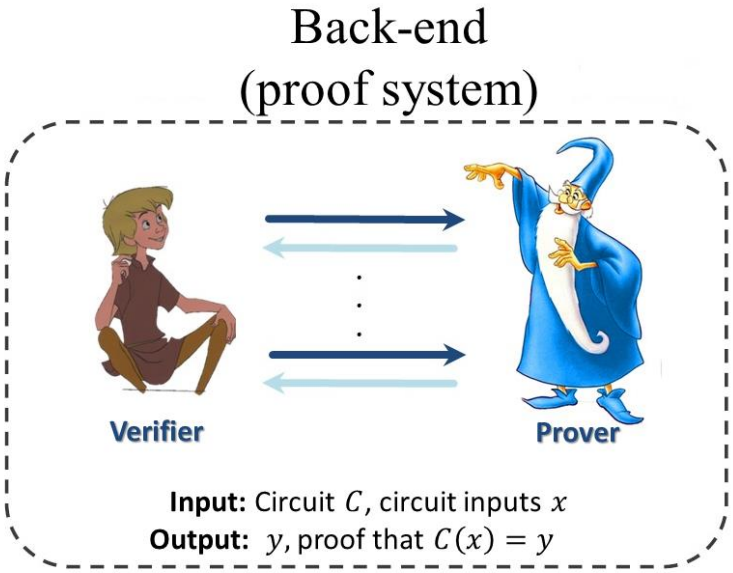
Input: Circuit C, circuit inputs x
Output: y, proof that $C(x) = y$

Usually, back-end only supports circuits over **finite fields…**

… which becomes a limitation for the front-end.

Doubly Efficient Interactive Proofs: Beyond finite fields

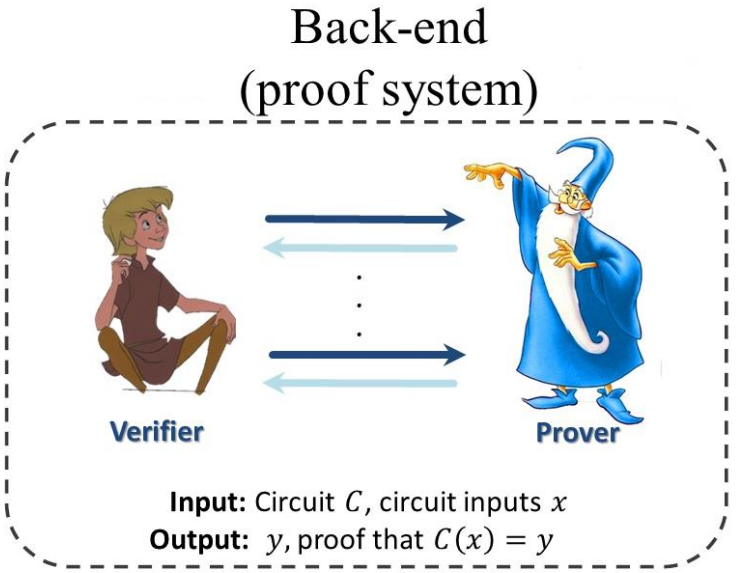
Usually, back-end only supports circuits over **finite fields…**



Doubly Efficient Interactive Proofs: Beyond finite fields

Usually, back-end only supports circuits over **finite fields…**

[CCKP19] extends to finite commutative rings.



[CCKP19] Chen, Cheon, Kim, and Park. "Verifiable computing for approximate computation" ePrint 2019/762.

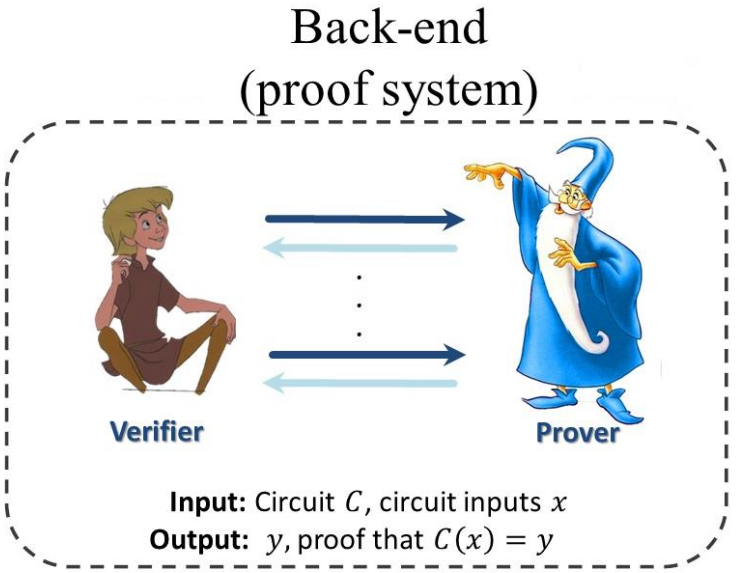
Doubly Efficient Interactive Proofs: Beyond finite fields

Usually, back-end only supports circuits over **finite fields…**

[CCKP19] extends to finite commutative rings.

This work: Extends to circuits over

- **Non-commutative rings**:
  Matrices, quaternions, Clifford algebras…
- **Infinite rings**:
  Integers ($\mathbb{Z}$), Reals ($\mathbb{R}$), Polynomial Rings…



[CCKP19] Chen, Cheon, Kim, and Park. "Verifiable computing for approximate computation" ePrint 2019/762.

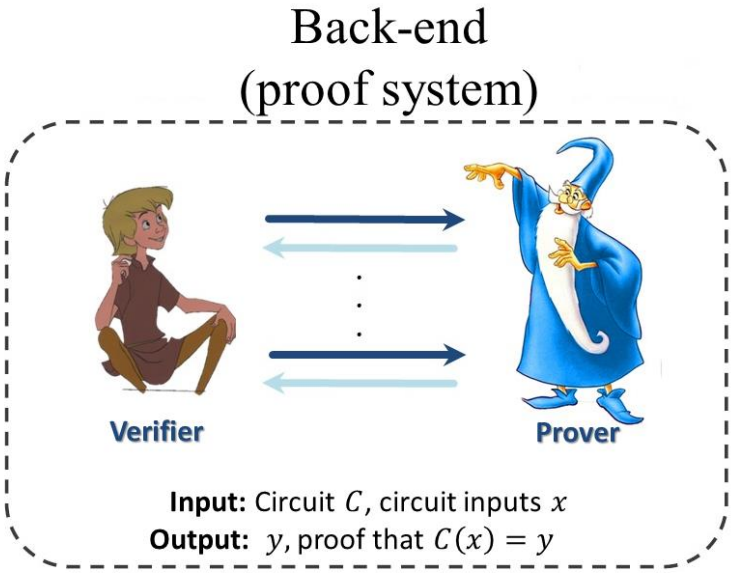
Doubly Efficient Interactive Proofs: Beyond finite fields

Usually, back-end only supports circuits over **finite fields…**

[CCKP19] extends to finite commutative rings.

This work: Extends to circuits over

- **Non-commutative rings**:
  Matrices, quaternions, Clifford algebras…
- **Infinite rings**:
  Integers ($\mathbb{Z}$), Reals ($\mathbb{R}$), Polynomial Rings…



We **preserve** or **improve** the complexity of the SotA finite-field-back-end

[CCKP19] Chen, Cheon, Kim, and Park. "Verifiable computing for approximate computation" ePrint 2019/762.

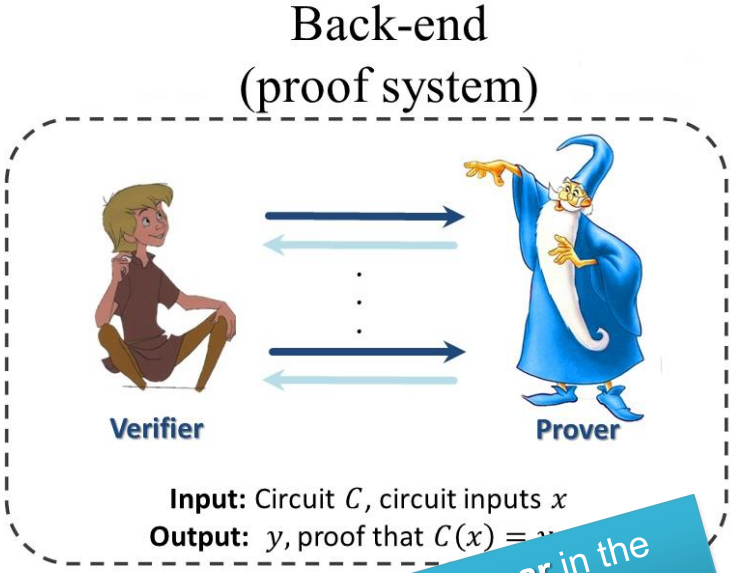
Doubly Efficient Interactive Proofs: Beyond finite fields

Usually, back-end only supports circuits over **finite fields…**

[CCKP19] extends to finite commutative rings.

This work: Extends to circuits over

- **Non-commutative rings**:
  Matrices, quaternions, **Clifford algebras**…
- **Infinite rings**:
  Integers ($\mathbb{Z}$), Reals ($\mathbb{R}$), **Polynomial Rings**…



We **preserve** or **improve** the complexity of the SotA finite-field-back-end

Our **prover** is actually **sublinear** in the size of the arithmetic circuit in some cases!!

[CCK… ark.
…or approximate
ePrint 2019/762.

Doubly Efficient Interactive Proofs: Beyond finite fields

Usually, back-end only supports circuits over **finite fields…**

[CCKP19] extends to finite commutative rings.

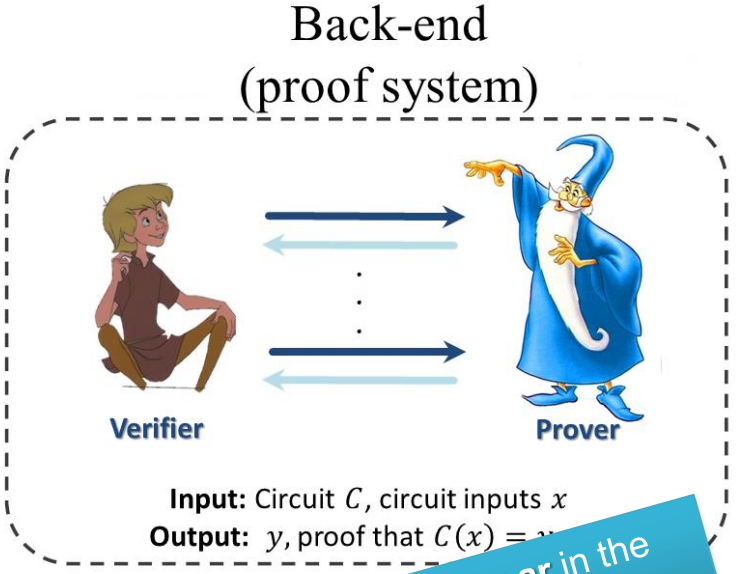
This work: Extends to circuits over

- **Non-commutative rings**:
  Matrices, quaternions, **Clifford algebras**…
- **Infinite rings**:
  Integers ($\mathbb{Z}$), Reals ($\mathbb{R}$), **Polynomial Rings**…

↯

IEEE 754 floating point numbers.

We **preserve** or **improve** the complexity of the SotA finite-field-back-end



Our **prover** is actually **sublinear** in the size of the arithmetic circuit in some cases!!

[CC… …ark.
… approximate
… ePrint 2019/762.

Exact vs Approximate Computation

No prior back-end supported infinite rings (e.g. $\mathbb{R}$) $\Rightarrow$ Approximate arithmetic (e.g. floating point).

Exact vs Approximate Computation

No prior back-end supported infinite rings (e.g. $\mathbb{R}$) ⇒ Approximate arithmetic (e.g. floating point).

ISSUES:

Approx. arith. does ***not*** constitute a ring:
Neither ***associative*** nor ***distributive***!
I.e. $a \cdot (b \cdot c) \neq (a \cdot b) \cdot c; a \cdot (b + c) \neq ab + ac$

⇒

Approx. arithmetic has to be emulated over some ring, which increases # gates.

Exact vs Approximate Computation

No prior back-end supported infinite rings (e.g. $\mathbb{R}$) ⇒ Approximate arithmetic (e.g. floating point).

ISSUES:

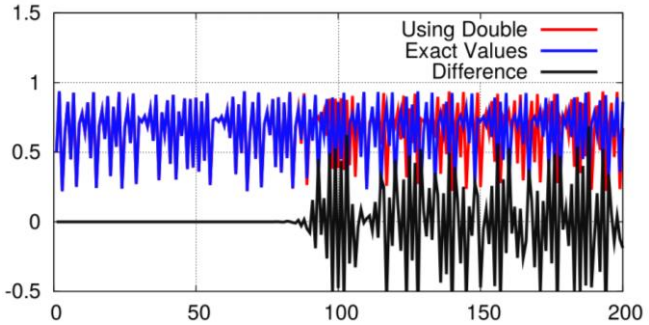
Approx. arith. does ***not*** constitute a ring:
Neither ***associative*** nor ***distributive***!
I.e. $a \cdot (b \cdot c) \neq (a \cdot b) \cdot c; a \cdot (b + c) \neq ab + ac$

⇒

Approx. arithmetic has to be emulated over some ring, which increases # gates.

Benign errors:

Relative error can be significant.



Logistic map: Double precision vs exact arithmetic.

$x_0 = 0.5;\ x_{n+1} = 3.75 \cdot x_n \cdot (1 - x_n).$

Exact vs Approximate Computation

No prior back-end supported infinite rings (e.g. $\mathbb{R}$) ⇒ Approximate arithmetic (e.g. floating point).

ISSUES:

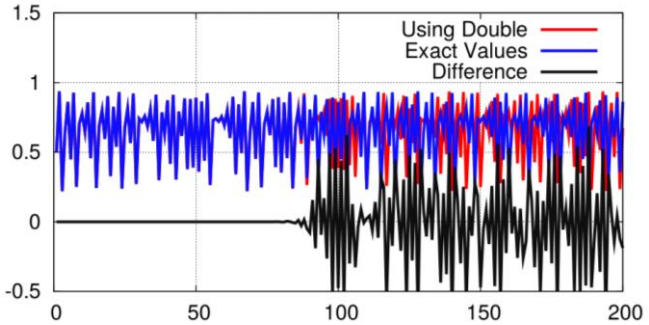
Approx. arith. does ***not*** constitute a ring:
Neither ***associative*** nor ***distributive***!
I.e. $a \cdot (b \cdot c) \neq (a \cdot b) \cdot c; a \cdot (b + c) \neq ab + ac$

⇒

Approx. arithmetic has to be emulated over some ring, which increases # gates.

Benign errors:

Relative error can be significant.



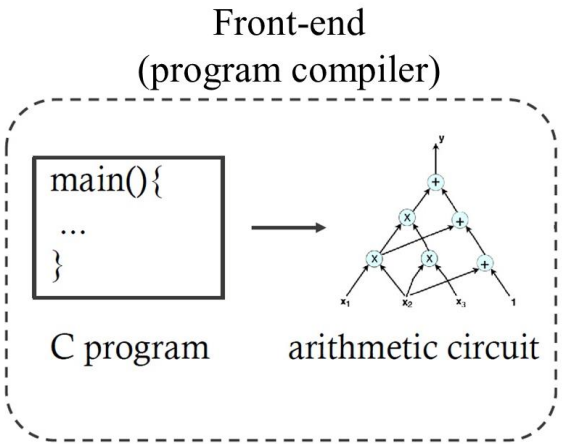
Logistic map: Double precision vs exact arithmetic.

$x_0 = 0.5;\ x_{n+1} = 3.75 \cdot x_n \cdot (1 - x_n).$

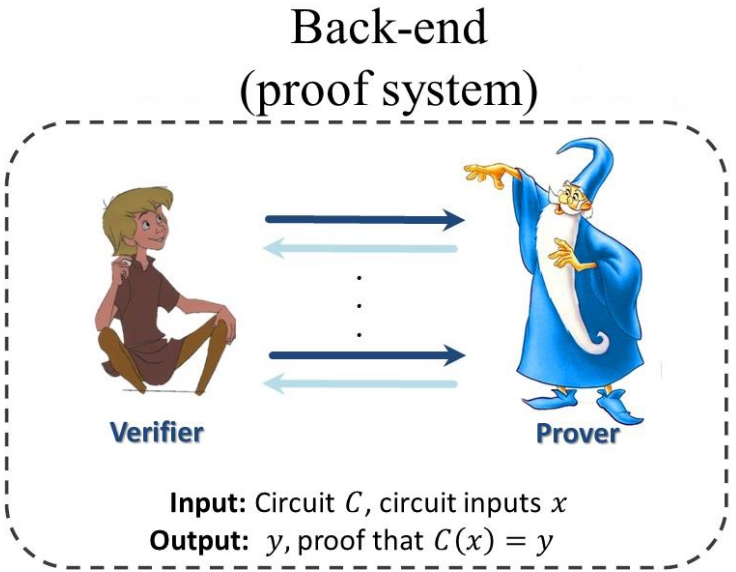
Malicious errors:

Circuit description biases the output.

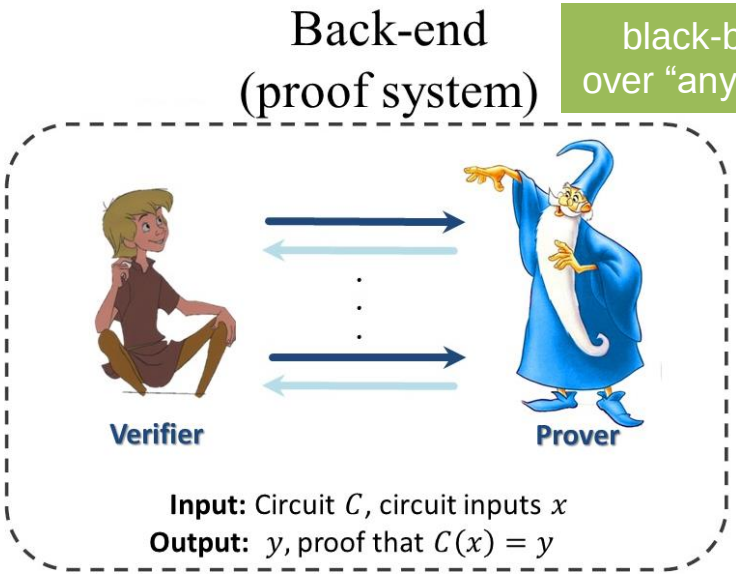
How do we agree on the front-end?



Contribution recap

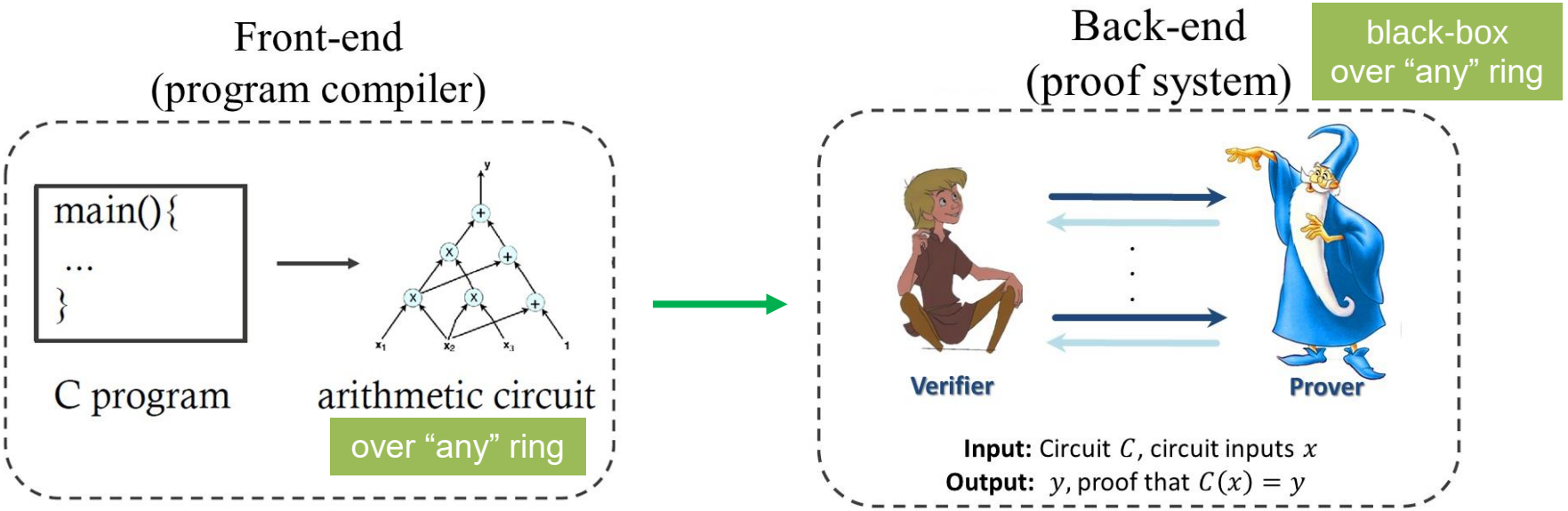


Contribution recap



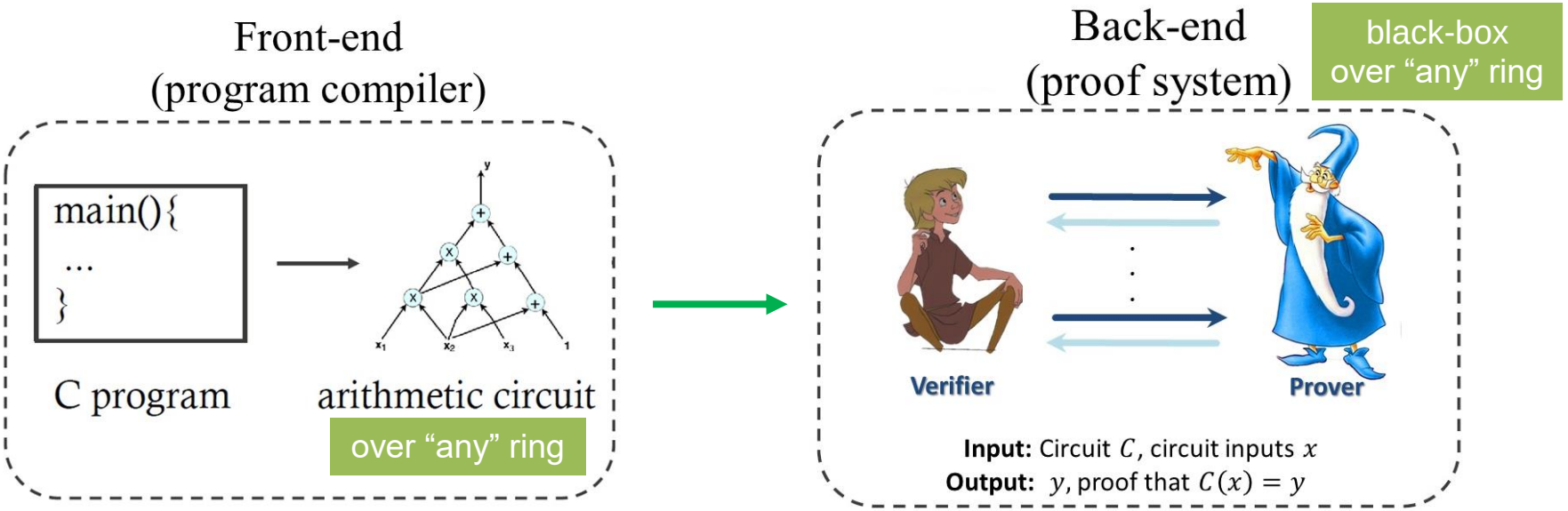
✓ **Theoretical** interest: Remove algebraic limitations, new rings with ***sublinear*** provers.

Contribution recap



- ✓ **Theoretical** interest: Remove algebraic limitations, new rings with ***sublinear*** provers.
- ✓ ***Agile*** software*:* Simpler front-end, ring-specific libraries can be reused.

Contribution recap



- ✓ **Theoretical** interest: Remove algebraic limitations, new rings with ***sublinear*** provers.
- ✓ ***Agile*** software*:* Simpler front-end, ring-specific libraries can be reused.
- ✓ Enable ***exact computation***: Impossible before, approx. arithmetic is an attack vector!

Technology Innovation Institute

Technical details sneak peek

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d-1.

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree **d-1**.

$0 = \mathrm{f}(a_2)$

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d-1.

$0 = \mathrm{f}(a_2) = \mathrm{q}(\mathbf{a_2}) \cdot (a_2 - a_1)$

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \ldots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d**-1**.

$0 = \mathrm{f}(a_2) = \mathrm{q}(a_2) \cdot (a_2 - a_1) \Rightarrow a_2$ is a root of q, **at most d-1 of those**.

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d-1.

$0 = \mathrm{f}(a_2) = \mathrm{q}(a_2) \cdot (a_2 - a_1) \Rightarrow a_2$ is a root of q, **at most d-1 of those**.

Euclidean division for polynomials $f \in R[X]$

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d-1.

$0 = \mathrm{f}(a_2) = \mathrm{q}(a_2) \cdot (a_2 - a_1) \Rightarrow a_2$ is a root of q, **at most d-1 of those**.

Euclidean division for polynomials $f \in R[X]$

Polynomial evaluation $Ev_a: R[X] \to R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Proof sketch (over a field): Induction on the degree d.

[…]

Let $a_1, a_2 \in A$ be two different **roots of f**. Then:

We have that $\mathrm{f(X)} = \mathrm{q(X)} \cdot (X - a_1)$, where $\mathrm{q(X)}$ is of degree d-1.

$0 = \mathrm{f}(a_2) = \mathrm{q}(a_2) \cdot (a_2 - a_1) \Rightarrow a_2$ is a root of q, **at most d-1 of those**.

Euclidean division for polynomials $f \in R[X]$

Polynomial evaluation $Ev_a: R[X] \to R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

$a_2 - a_1$ is not a zero divisor ($\forall$ pair a_i, a_j $i \neq j$)

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Euclidean division for polynomials $f \in R[X]$

Polynomial evaluation $Ev_a: R[X] \rightarrow R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

$a_2 - a_1$ is not a zero divisor ($\forall$ pair a_i, a_j $i \neq j$)

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Euclidean division for polynomials $f \in R[X]$

Polynomial evaluation $Ev_a: R[X] \to R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

$a_2 - a_1$ is not a zero divisor ($\forall$ pair a_i, a_j $i \neq j$)

Assumption on $\mathbf{A}$ (“regular difference set”)

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Euclidean division for polynomials $f \in R[X]$

Easily follow if $\mathbf{A} \subseteq Z(R)$
(i.e.$\forall a \in A, r \in R, a \cdot r = r \cdot a$)

Polynomial evaluation $Ev_a: R[X] \to R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

$a_2 - a_1$ is not a zero divisor
($\forall$ pair a_i, a_j $i \neq j$)

Assumption on $\mathbf{A}$
(“regular difference set”)

“Any” ring R

black-box over “any” ring

R contains a big enough set $\mathbf{A} = \{a_1, \dots a_n\}$ such that:
$\forall f \in R[X]$ of degree d, f has at most d roots in $\mathbf{A}$.

Euclidean division for polynomials $f \in R[X]$

Easily follow if $\mathbf{A} \subseteq Z(R)$
(i.e.$\forall a \in A, r \in R, a \cdot r = r \cdot a$)

What happens if we relax $\mathbf{A}$ to commutative set?
(i.e.$\forall a_i, a_j \in A, a_i \cdot a_j = a_j \cdot a_i$)

Polynomial evaluation $Ev_a: R[X] \to R$ at any $\mathbf{a} \in \mathbf{A}$ is a ring homomorphism

$a_2 - a_1$ is not a zero divisor
($\forall$ pair a_i, a_j $i \neq j$)

Assumption on $\mathbf{A}$
(“regular difference set”)

The polynomial ring R[X] for non-commutative R

For a commutative set $\mathbf{A}$, we define a new polynomial ring $\boldsymbol{R[X]}$

Euclidean division
for polynomials $\boldsymbol{f \in R[X]}$

Polynomial evaluation
$\boldsymbol{Ev_a: R[X] \to R}$ at any $\mathbf{a \in A}$
is a ring homomorphism

The polynomial ring R[X] for non-commutative R

For a commutative set $\mathbf{A}$, we define a new polynomial ring $\boldsymbol{R[X]}$

- **Core idea:** $\boldsymbol{X}$ commutes with whatever the set $\mathbf{A}$ does.

Euclidean division
for polynomials $\boldsymbol{f \in R[X]}$

Polynomial evaluation
$\boldsymbol{Ev_a: R[X] \to R}$ at any $\mathbf{a \in A}$
is a ring homomorphism

The polynomial ring R[X] for non-commutative R

For a commutative set $\mathbf{A}$, we define a new polynomial ring $\boldsymbol{R[X]}$

- **Core idea:** $\boldsymbol{X}$ commutes with whatever the set $\mathbf{A}$ does.
- Actually, we can only bound roots in $\mathbf{A}$ for a ***subset*** of polys in our new $\boldsymbol{R[X]}$.

Euclidean division
for polynomials $\boldsymbol{f \in R[X]}$

Polynomial evaluation
$\boldsymbol{Ev_a: R[X] \to R}$ at any $\mathbf{a \in A}$
is a ring homomorphism

The polynomial ring R[X] for non-commutative R

For a commutative set $\mathbf{A}$, we define a new polynomial ring $\boldsymbol{R[X]}$

- **Core idea:** $\boldsymbol{X}$ commutes with whatever the set $\mathbf{A}$ does.
- Actually, we can only bound roots in $\mathbf{A}$ for a ***subset*** of polys in our new $\boldsymbol{R[X]}$.
- Need to modify layer-consistency equation in GKR and sumcheck so as to always encounter polynomials in the right subset.

Euclidean division
for polynomials $\boldsymbol{f \in R[X]}$

Polynomial evaluation
$\boldsymbol{Ev_a: R[X] \to R}$ at any $\mathbf{a \in A}$
is a ring homomorphism

The polynomial ring R[X] for non-commutative R

For a commutative set $\mathbf{A}$, we define a new polynomial ring $\boldsymbol{R[X]}$

- **Core idea:** $\boldsymbol{X}$ commutes with whatever the set $\mathbf{A}$ does.
- Actually, we can only bound roots in $\mathbf{A}$ for a ***subset*** of polys in our new $\boldsymbol{R[X]}$.
- Need to modify layer-consistency equation in GKR and sumcheck so as to always encounter polynomials in the right subset.
- We provide a linear-time algorithm for the prover.

Euclidean division
for polynomials $\boldsymbol{f \in R[X]}$

Polynomial evaluation
$\boldsymbol{Ev_a: R[X] \to R}$ at any $\mathbf{a \in A}$
is a ring homomorphism

Contribution recap

Thm: Let **R** be a (possibly infinite, possibly non-commutative) ring containing a **commutative**, **regular difference set A**.
There exists a doubly-efficient IP for layered arithmetic circuits over **R**.

Contribution recap

Thm: Let **R** be a (possibly infinite, possibly non-commutative) ring containing a **commutative**, **regular difference set A**.
There exists a doubly-efficient IP for layered arithmetic circuits over **R**.

- ✓ **Theoretical** interest: Remove algebraic limitations, new rings with ***sublinear*** provers.
- ✓ ***Agile*** software*:* Simpler front-end, ring-specific libraries can be reused.
- ✓ Enable ***exact computation***: Impossible before, approx. arithmetic is an attack vector!

Technology Innovation Institute

Thanks for your attention!

Questions?

Full version: ia.cr/2022/587

Eduardo Soria-Vazquez