Proofs of Work From Worst-Case Assumptions

Marshall Ball — Columbia University / IDC Herzliya

Alon Rosen — IDC Herzliya

Manuel Sabin — UC Berkeley

Prashant Vasudevan — MIT → UC Berkeley

DISCLAIMER:

THIS TALK IS

NOT

ABOUT BITCOIN.

A Cryptographer's Dream Theorem

interesting Cryptography

OR

efficient SAT algorithms

A Cryptographer's Dream Theorem

interesting Cryptography

OR

efficient worst-case SAT algorithms

A Cryptographer's Dream Theorem

interesting Cryptography

OR

efficient worst-case SAT algorithms

↕

rich theory of NP Completeness

THIS WORK

"THEOREM"

interesting cryptography

OR

non-trivial SAT speed ups

THIS WORK

"THEOREM"

interesting cryptography

OR

non-trivial worst-case SAT speed ups

THIS WORK

"THEOREM"

interesting cryptography

OR

non-trivial worst-case SAT speed ups

rich theory of fine-grained complexity

Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

T: "amount of work"

T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

Prover — T: "amount of work" — Verifier

T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

Prover — T: "amount of work" — Verifier



T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

Prover — T: "amount of work" — Verifier



Prove(c)



T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

Prover — T: "amount of work" — Verifier

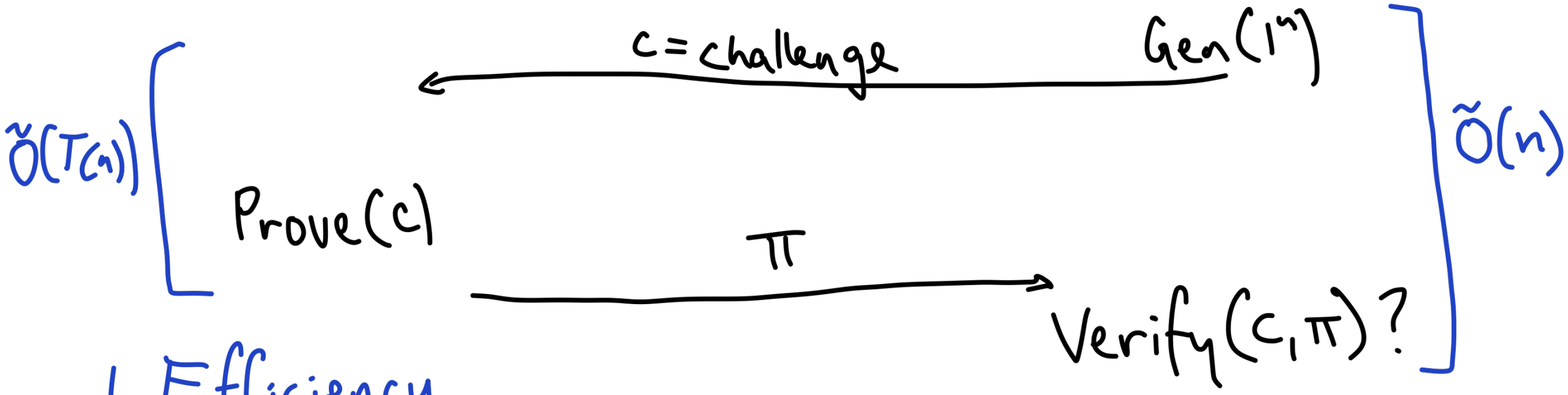




T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

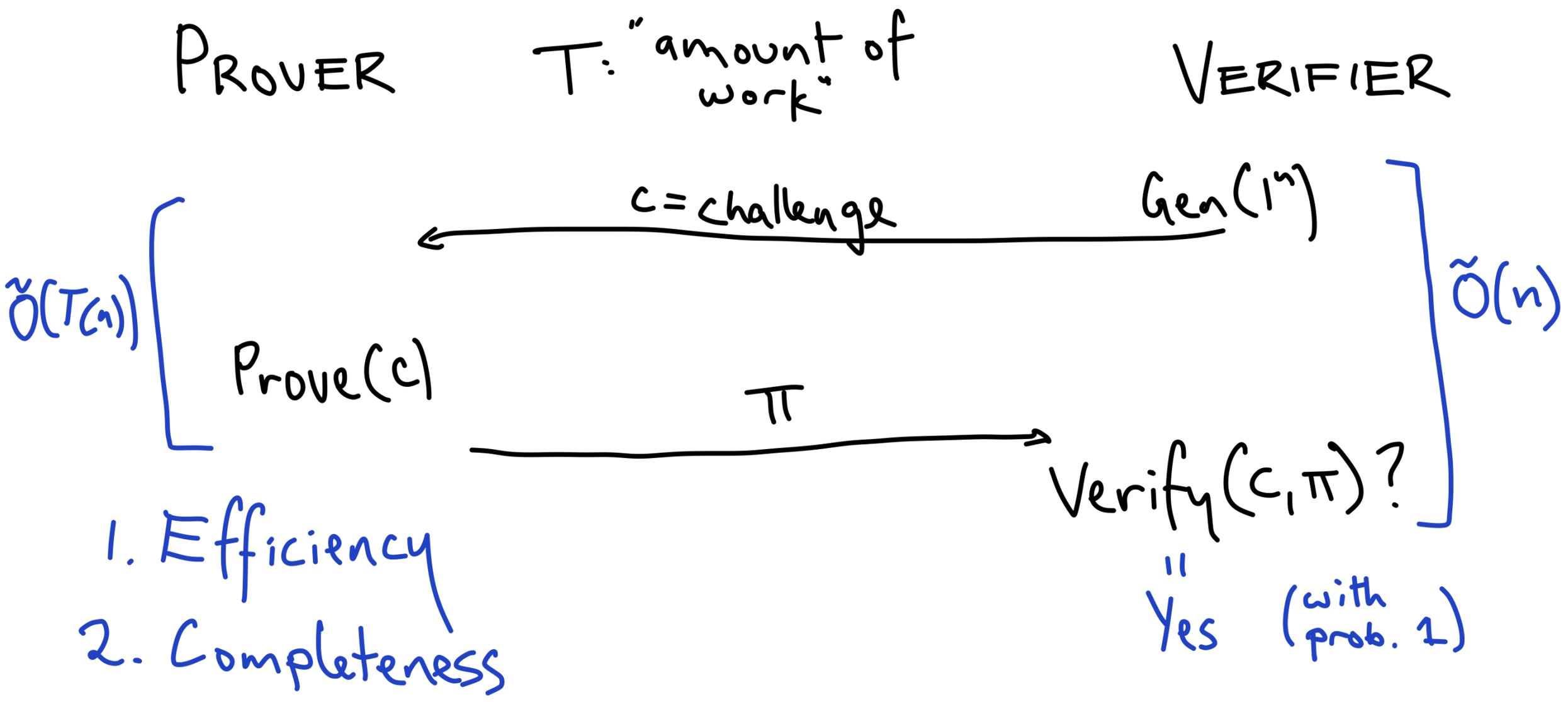
Prover — T: "amount of work" — Verifier



1. Efficiency

T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

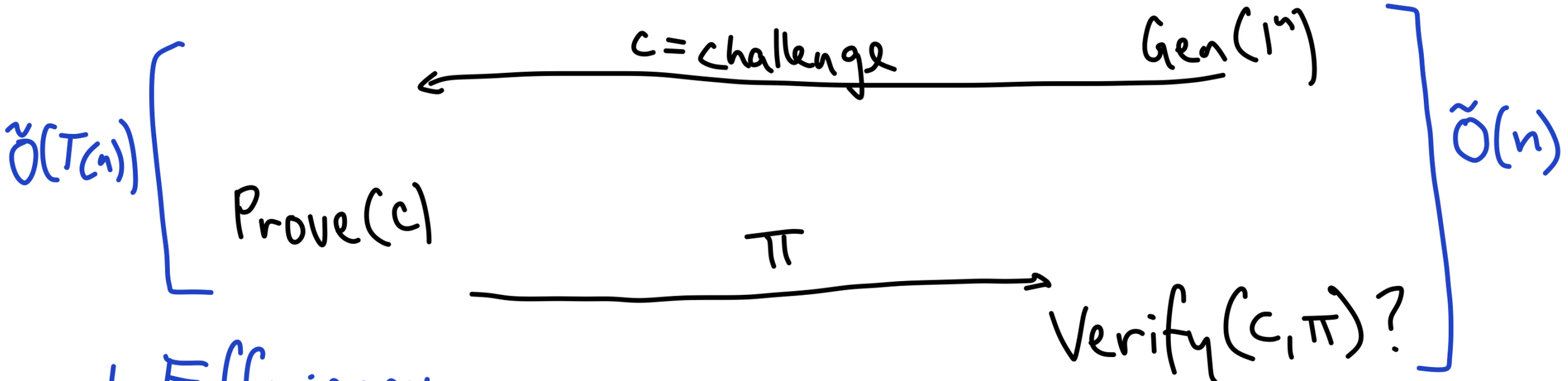


1. Efficiency
2. Completeness

T-Proof of Work

[Dwork Naor '92, Jakobsson Juels '99]

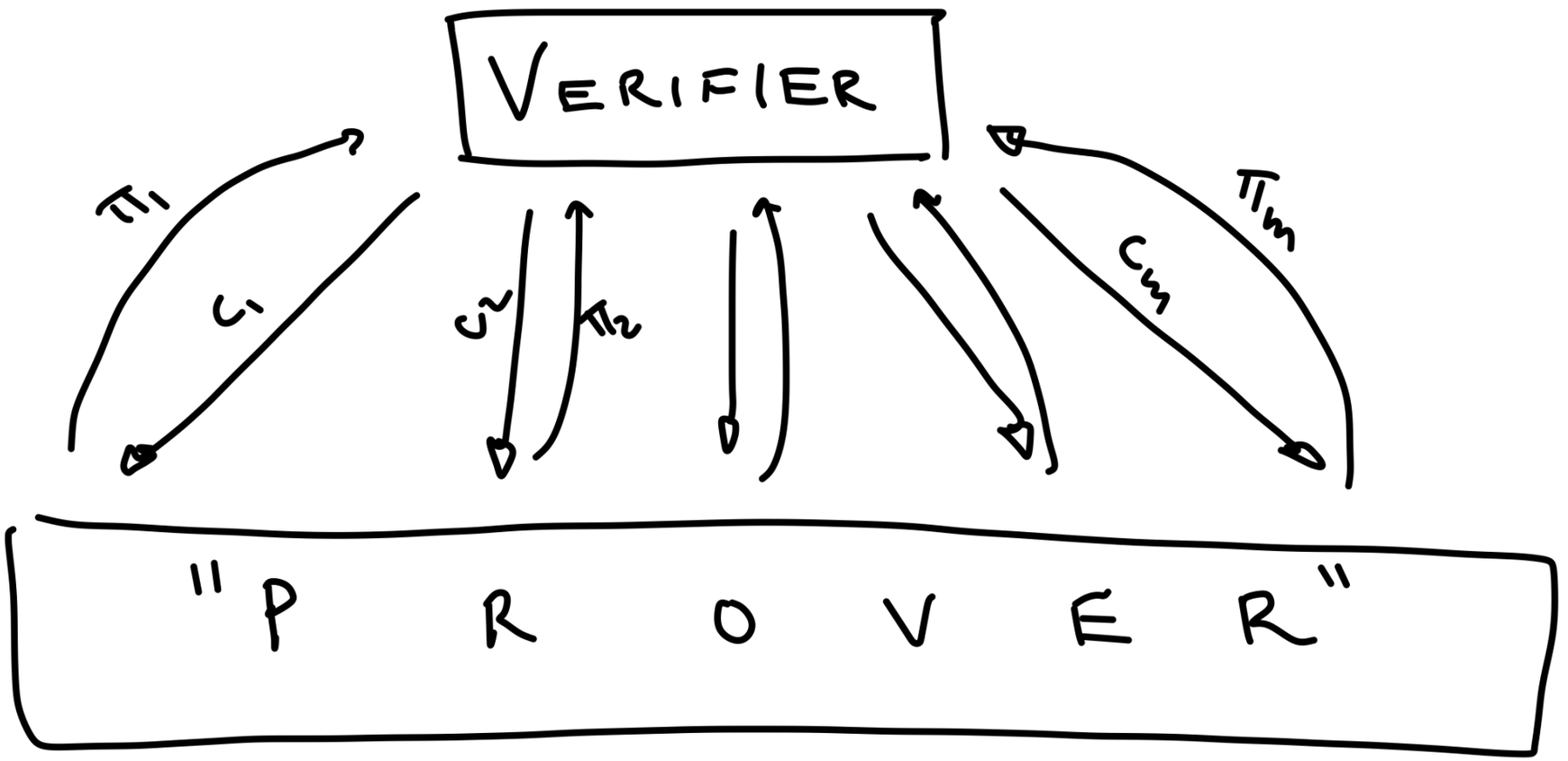
Prover — T: "amount of work" — Verifier



1. Efficiency
2. Completeness
3. Hardness: "T(n) is required to convince verifier"

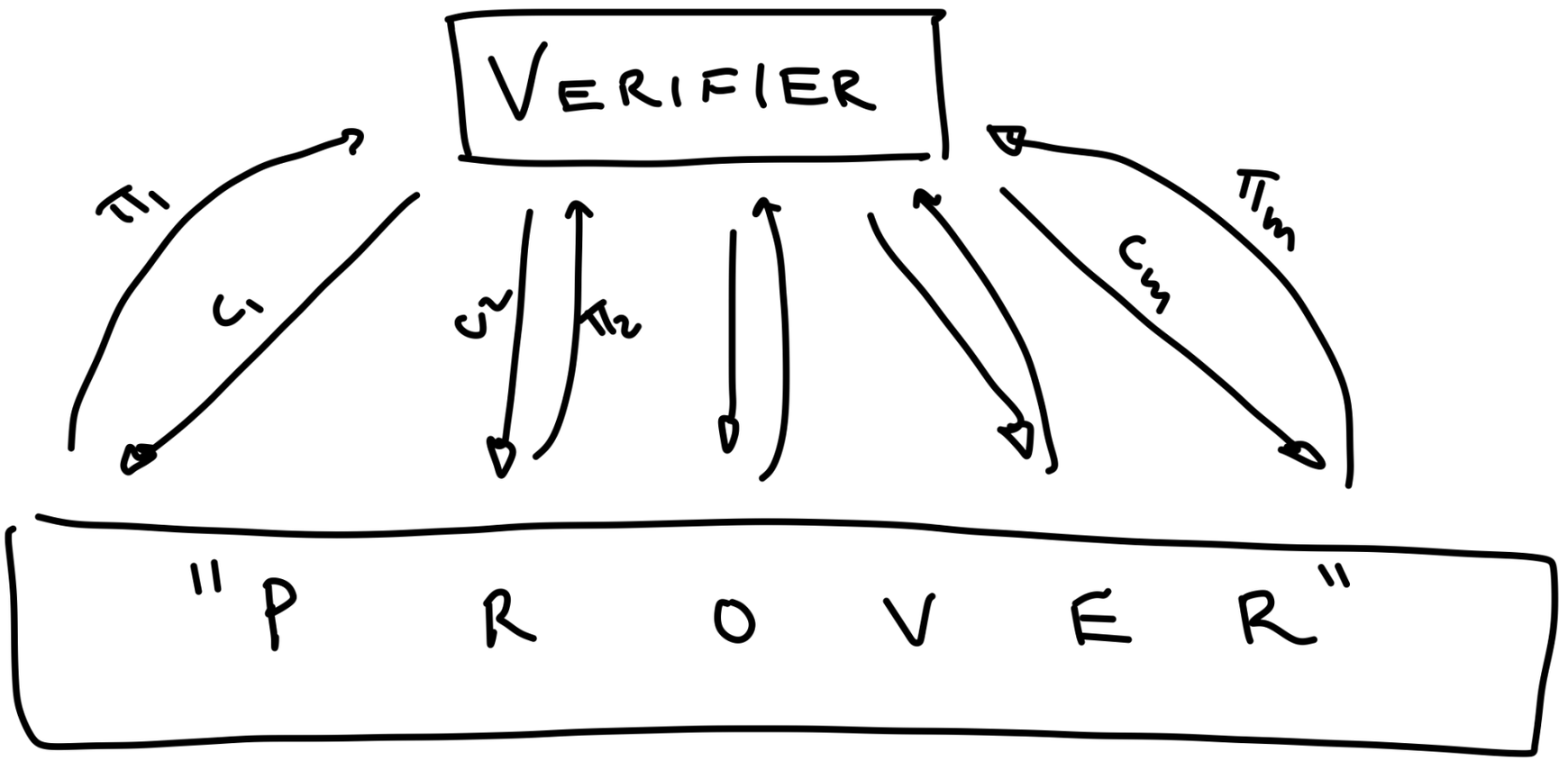
HARDNESS

(Non-Amortizable up to small factors)



HARDNESS

(Non-Amortizable up to small factors)



Given $m(n)$ challenges, batch processing $m(n)$ proofs in time $m(n)(T(n))^{1-\varepsilon}$ succeeds with probability $< 1/n^{\omega(1)}$ for any $\varepsilon > 0$.

Informal Mad-Lib Theorem

Assuming _____ ,
WORST-CASE CONJECTURE FROM FINE-GRAINED COMPLEXITY

then the "recipe" (to be alluded to)

yields a Proof of Work Scheme

requiring _____ work.
CONJECTURED HARDNESS

ROADMAP

ROADMAP

1

The Assumptions:
(some) Worst-Case Conjectures
from
Fine-Grained Complexity

Roadmap

1 The Assumptions:
(some) Worst-Case Conjectures
from
Fine-Grained Complexity

2 Specific Example:
Orthogonal Vectors

ROADMAP

1 The Assumptions:
(some) Worst-Case Conjectures
from
Fine-Grained Complexity

2 Specific Example:
Orthogonal Vectors

3 "Nice" Properties

FINE-GRAINED COMPLEXITY

FINE-GRAINED COMPLEXITY

Observation: for many natural problems, "brute force" is essentially state of the art.

FINE-GRAINED COMPLEXITY

Observation: for many natural problems, "brute force" is essentially state of the art.

"brute force": T

state of the art: $\Omega(T^{1-\varepsilon})\ \forall \varepsilon > 0$

FINE-GRAINED COMPLEXITY

Observation: for many natural problems,

"brute force" is essentially state of the art.

T

$\Omega(T^{1-\varepsilon})\ \forall \varepsilon > 0$

Theory: improving "exact" complexity of X

$\Downarrow$

improving "exact" complexity of Y

FINE-GRAINED COMPLEXITY

Observation: for many natural problems,

"brute force" is essentially state of the art.

T

$\Omega(T^{1-\varepsilon})$ $\forall \varepsilon > 0$

Theory: improving "exact" complexity of X

$\Downarrow$

improving "exact" complexity of Y

} fine-grained reductions

FINE-GRAINED COMPLEXITY

Observation: for many natural problems, "brute force" is essentially state of the art.

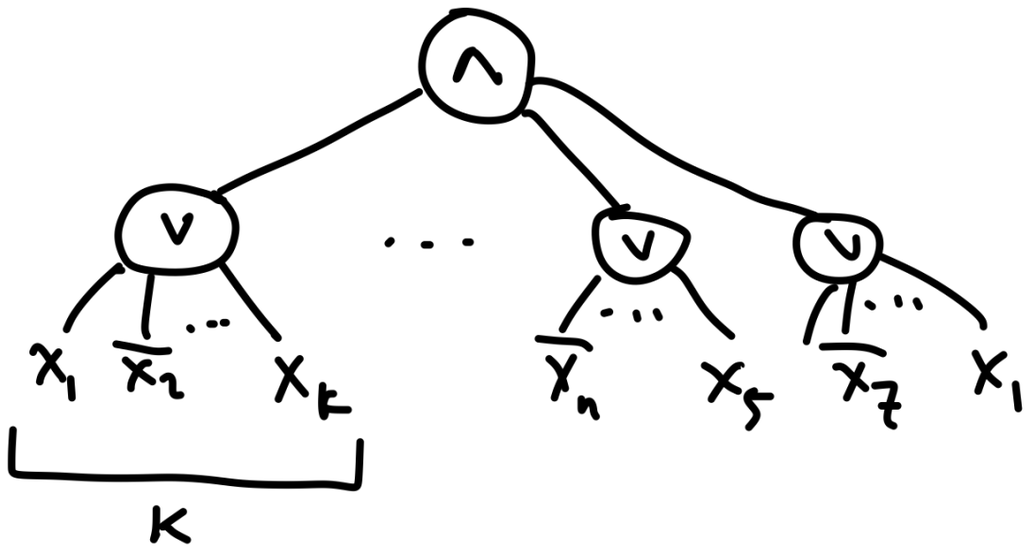
"brute force" $= T$

state of the art $= \Omega(T^{1-\varepsilon})\ \forall \varepsilon > 0$

Theory: improving "exact" complexity of X $\Rightarrow$ improving "exact" complexity of Y } fine-grained reductions

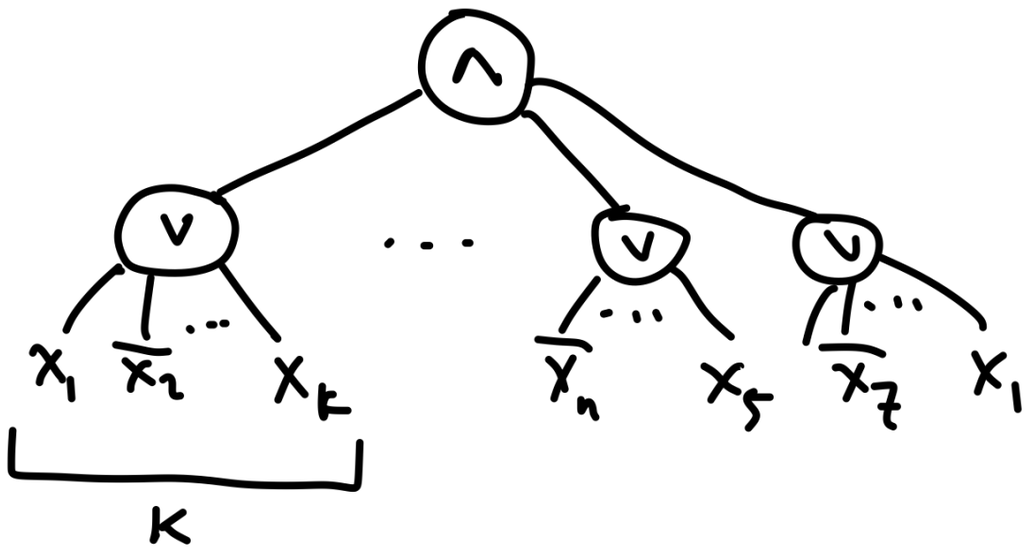
Conjectures: for many natural problems, brute force is essentially optimal.

K-SAT

is 

satisfiable?

k-SAT

is 

satisfiable?

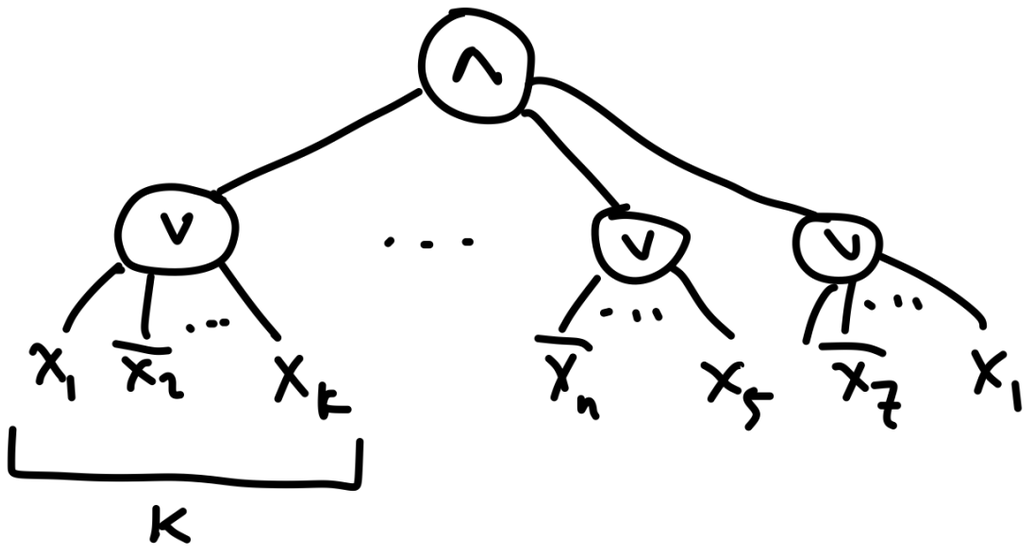
Best Worst case Algorithm

$$O\left(2^{(1-\Theta(1/k))n}\right)$$

[Paturi Pudlak Saks Zane '05]

k-SAT

is



satisfiable?

Best Worst case Algorithm

$$O\left(2^{(1-\Theta(1/k))n}\right)$$

[Paturi Pudlak Saks Zane '05]

(SETH)

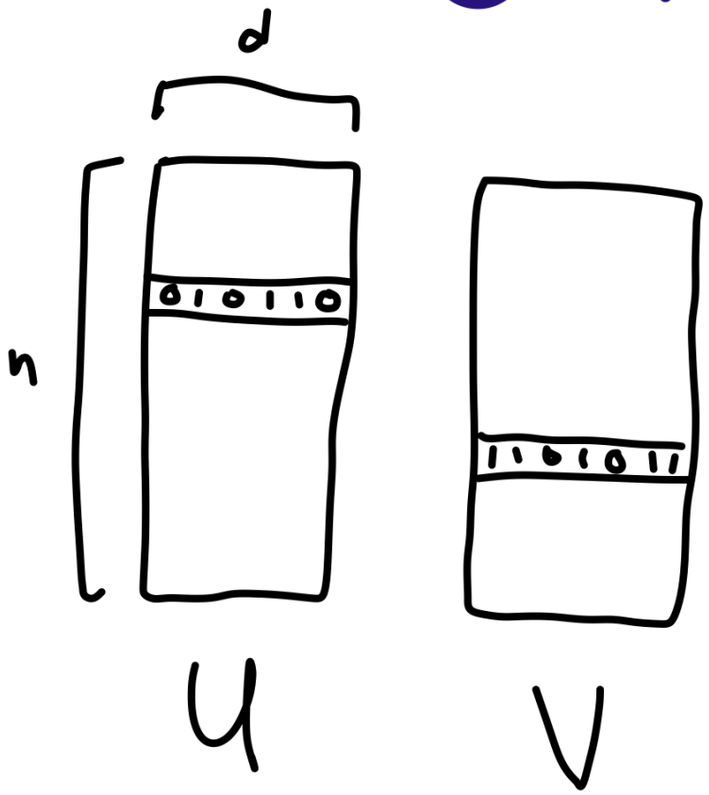
Strong Exponential Time Hypothesis

$\forall \varepsilon, \exists k$: kSAT cannot be solved in time

$$O\left(2^{(1-\varepsilon)n}\right)$$

[Impagliazzo Paturi '99]

Orthogonal Vectors

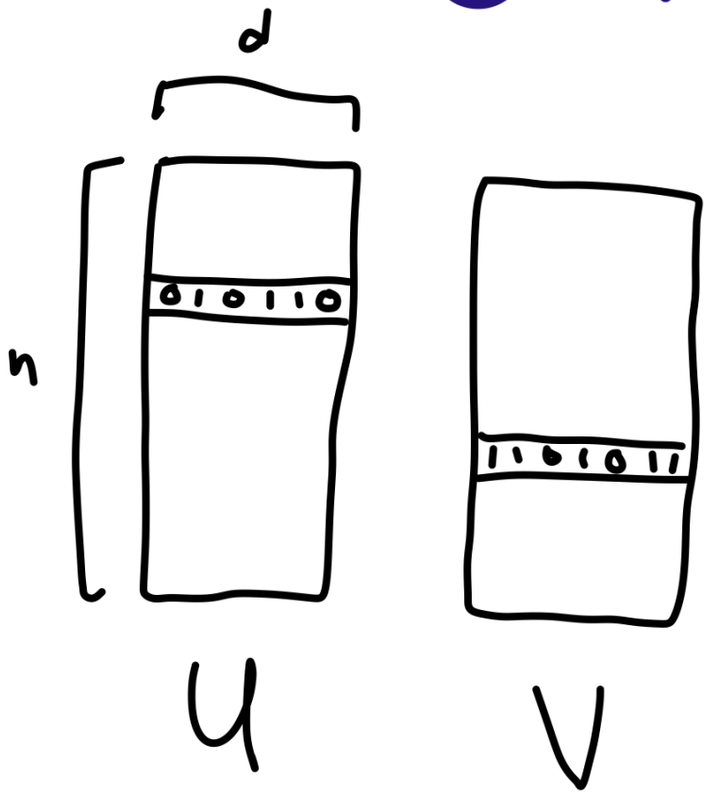


is there $u \in U, v \in V$:

$\langle u, v \rangle = 0$?

(u, v represent disjoint subsets of $[d]$?)

Orthogonal Vectors



is there $u \in U, v \in V$:

$\langle u, v \rangle = 0$?

(u, v represent disjoint subsets of $[d]$?)

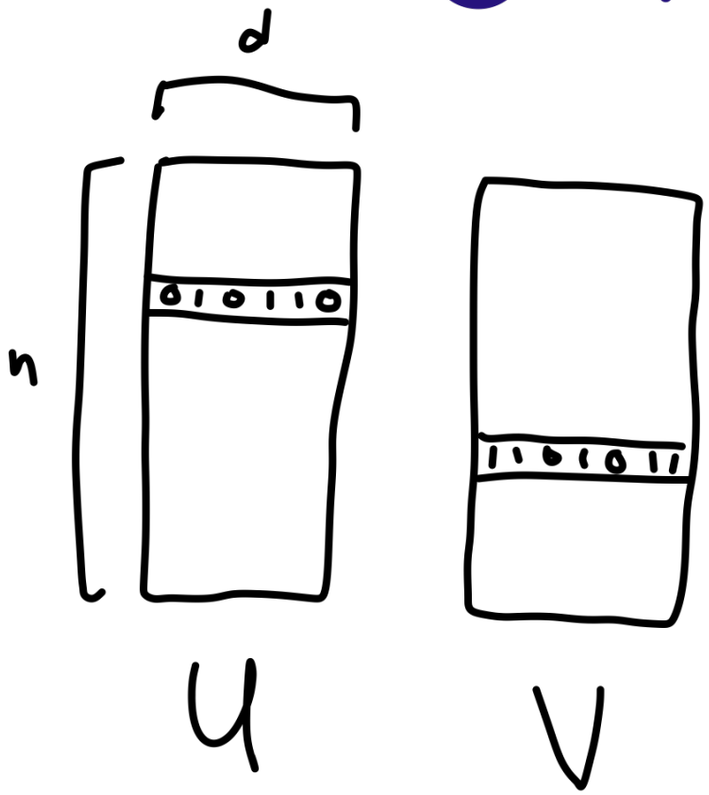
Best Algorithm

Worst case

$O(n^{2-\Theta(1/\log(d/\log n))})$

[AbboudWilliamsYu'15, ChanWilliams'16]

Orthogonal Vectors



is there $u \in U, v \in V$:

$\langle u, v \rangle = 0$?

(u, v represent disjoint subsets of $[d]$?)

Best Worst case Algorithm

$O(n^{2 - \Theta(1/\log(d/\log n))})$

[AbboudWilliamsYu'15, ChanWilliams'16]

Orthogonal Vectors Conjecture

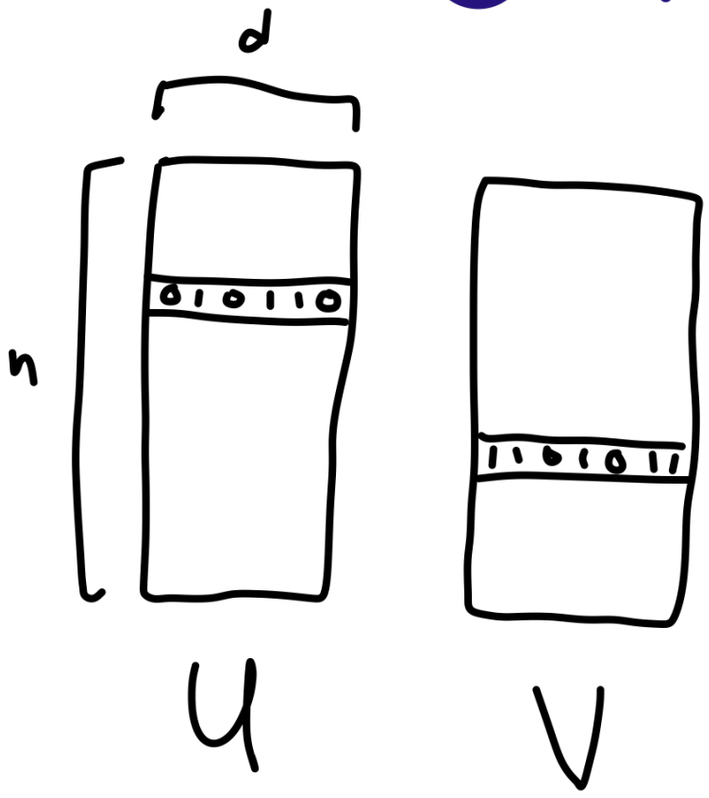
for $d = \omega(\log n)$, $\forall \varepsilon > 0$,

OV cannot be solved in time

$O(n^{2-\varepsilon})$

[Williams'05]

Orthogonal Vectors



is there $u \in U, v \in V$:

$\langle u, v \rangle = 0$?

(u, v represent disjoint subsets of $[d]$?)

SETH

Best Worst case Algorithm

$O(n^{2-\theta(1/\log(d/\log n))})$

[AbboudWilliamsYu'15, ChanWilliams'16]

Orthogonal Vectors Conjecture

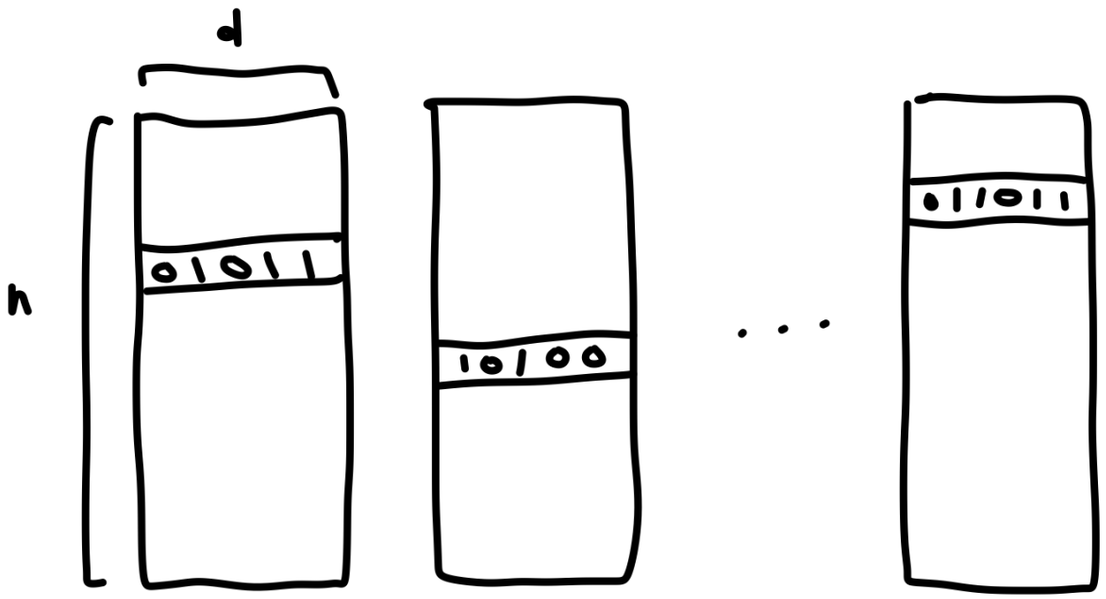
for $d = \omega(\log n)$, $\forall \varepsilon > 0$,

OV cannot be solved in time

$O(n^{2-\varepsilon})$

[Williams'05]

k-Orthogonal Vectors



is there

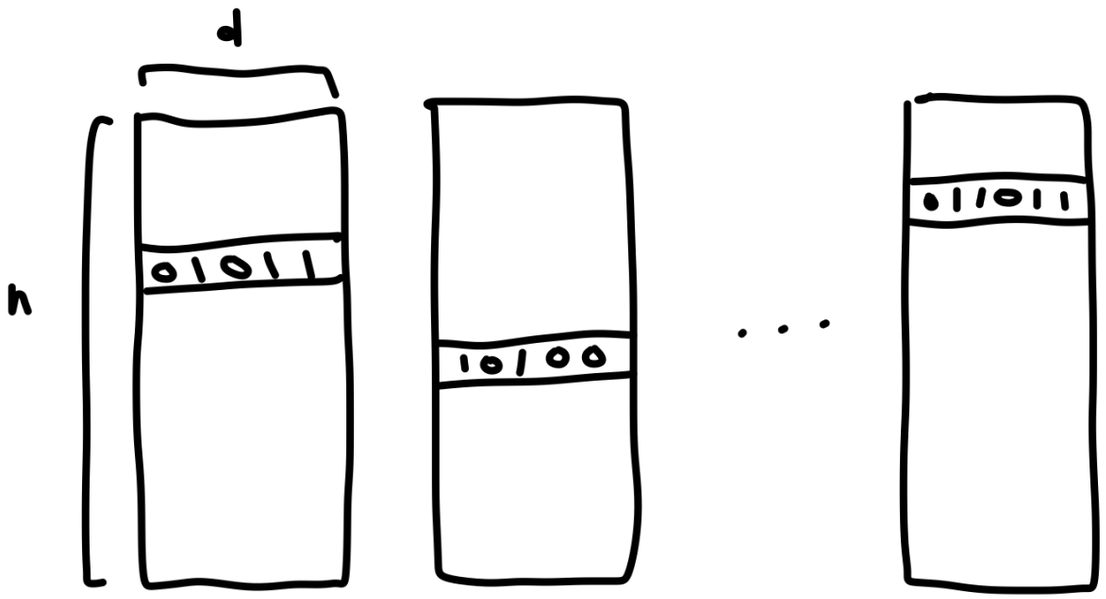
$u^{(1)} \in U_1, u^{(2)} \in U_2, \ldots, u^{(k)} \in U_k$

such that

$$\sum_{i=d}^{d} u_i^{(1)} u_i^{(2)} \cdots u_i^{(3)} = 0 \; ?$$

(is the intersection sets rep. by $u^{(j)}$'s empty?)

k-Orthogonal Vectors



is there

$u^{(1)} \in U_1, u^{(2)} \in U_2, \ldots, u^{(k)} \in U_k$

such that

$$\sum_{i=d}^{d} u_i^{(1)} u_i^{(2)} \cdots u_i^{(3)} = 0 \; ?$$

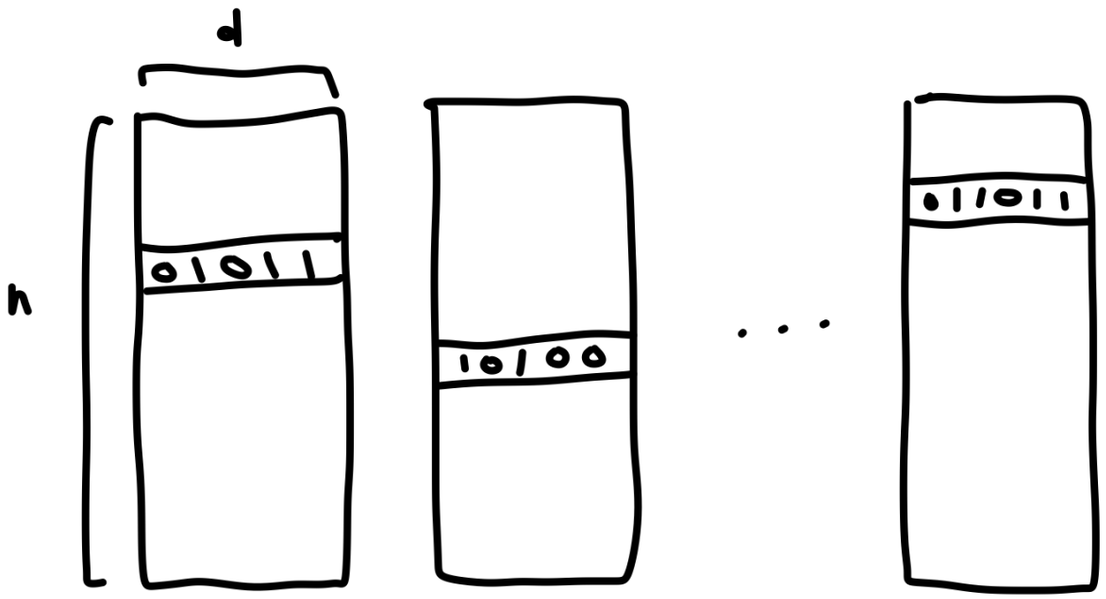
(is the intersection sets rep. by $u^{(i)}$'s empty?)

K-Orthogonal Vectors Conjecture

for $d = \omega(\log n)$, $\forall \varepsilon > 0$,

K-OV cannot be solved in time $O(n^{k-\varepsilon})$

k-Orthogonal Vectors



is there

$u^{(1)} \in U_1, u^{(2)} \in U_2, \ldots, u^{(k)} \in U_k$

such that

$$\sum_{i=d}^{d} u_i^{(1)} u_i^{(2)} \cdots u_i^{(3)} = 0 \; ?$$

(is the intersection sets rep. by $u^{(i)}$'s empty?)

K-Orthogonal Vectors Conjecture

for $d = \omega(\log n)$, $\forall \varepsilon > 0$,
K-OV cannot be solved in time $O(n^{k-\varepsilon})$

$\Leftarrow$ SETH

Many More

All-Pairs Shortest Paths

given G, find length of shortest paths between all nodes

conj. n^3-hard

3SUM

given $S \subseteq \{-n^3, \ldots, n^3\}$: $|S| = n$,

is there $a, b, c \in S$ s.t.

$a + b + c = 0$?

conj. n^2-hard

Zero Weight Triangle

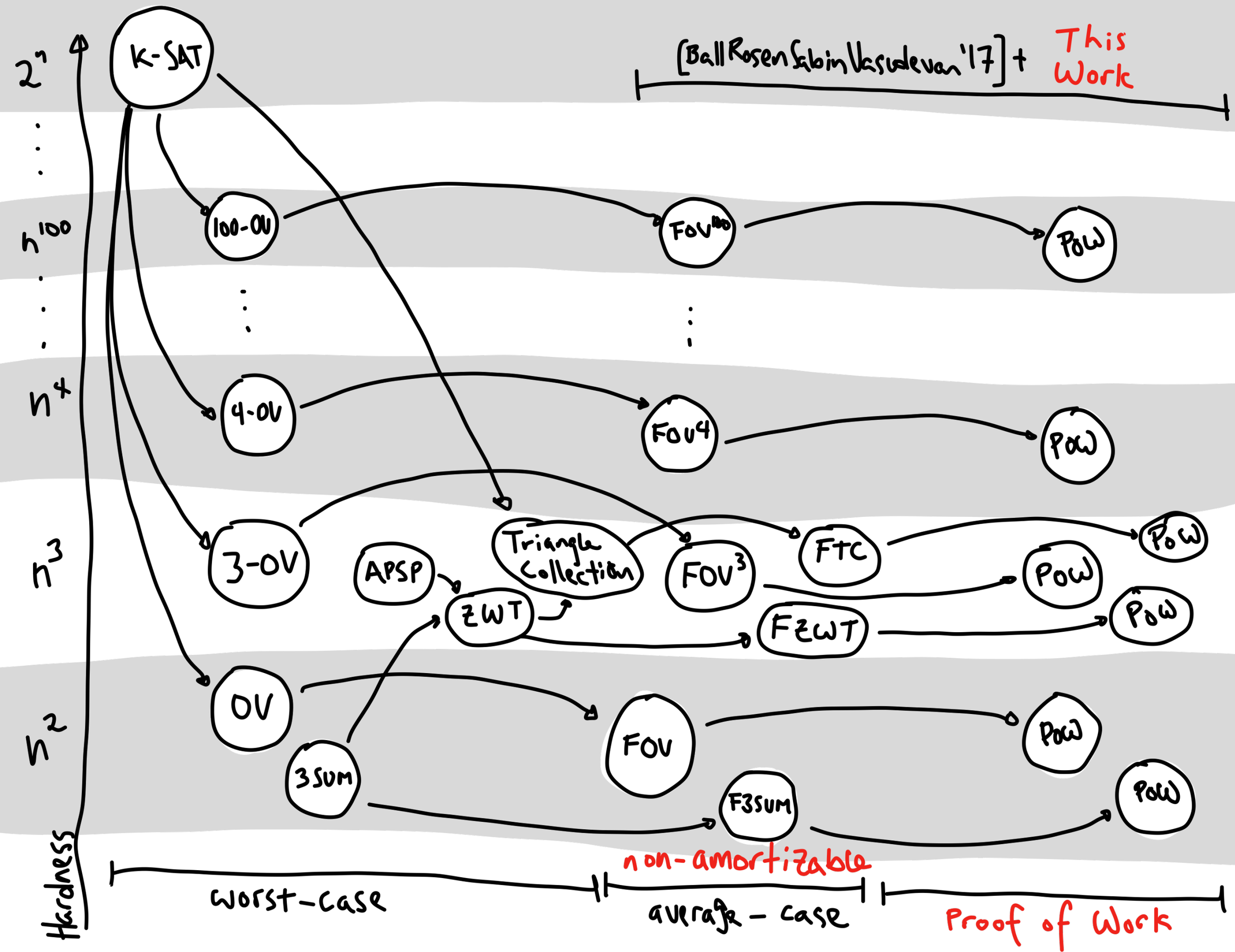
given edge-weighted G,

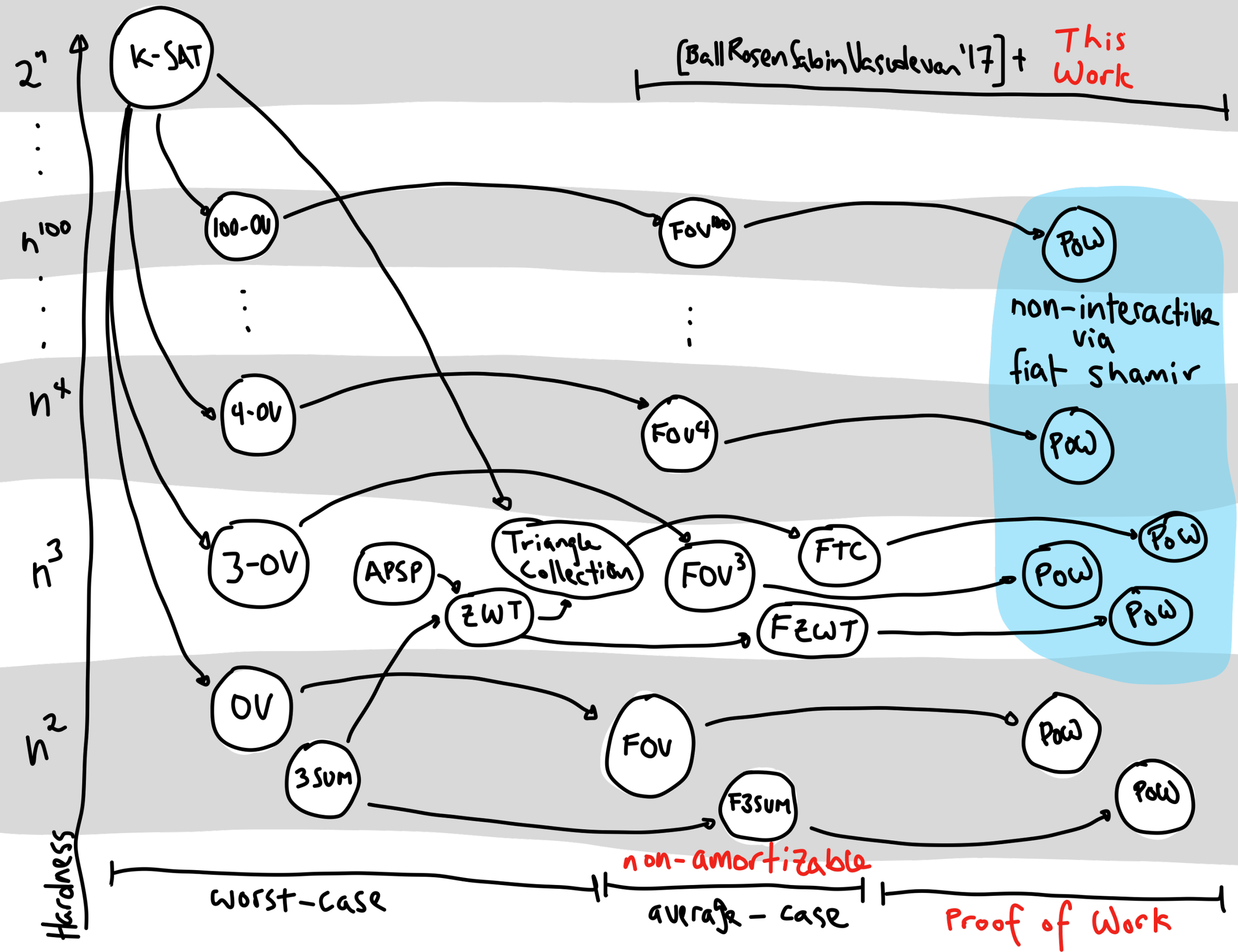
find triangle with weights

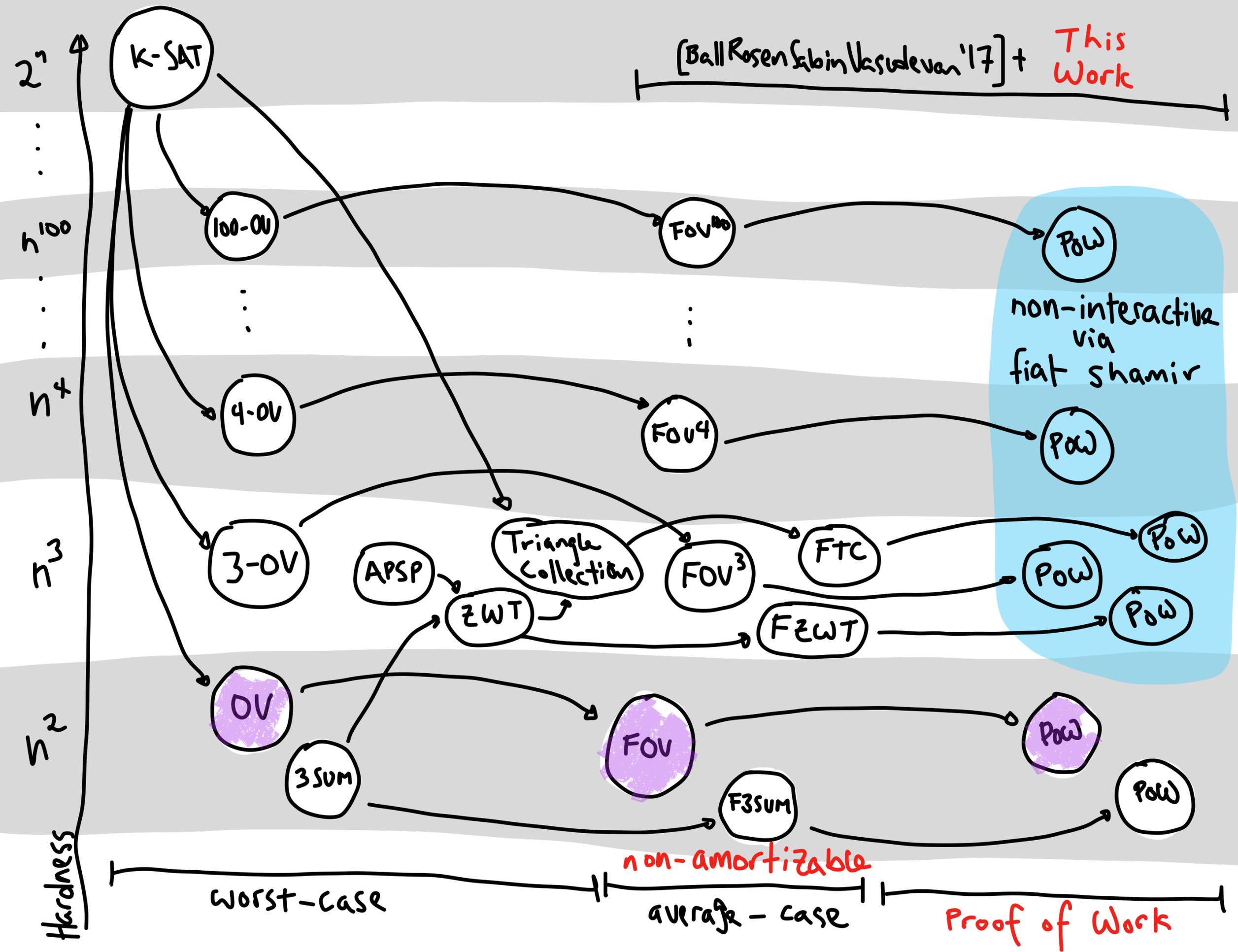
$w_i + w_j + w_k = 0$

conj. n^3-hard

etc, etc, ...


2^n
n^100
n^4
n^3
n^2
Hardness
K-SAT
100-OV
4-OV
3-OV
OV
APSP
3SUM
Triangle Collection
ZWT
[BallRosenSabinVasudevan'17] + This Work
FOV^100
FOV^4
FOV^3
FOV
FTC
FZWT
F3SUM
PoW
worst-case
non-amortizable
average-case
Proof of Work


2^n
n^{100}
n^4
n^3
n^2
Hardness
K-SAT
100-OV
4-OV
3-OV
OV
APSP
3SUM
Triangle Collection
ZWT
FOV^{100}
FOV^4
FOV^3
FOV
FTC
FZWT
F3SUM
PoW
non-interactive via fiat shamir
worst-case
non-amortizable
average-case
Proof of Work
[BallRosenSabinVasudevan'17] + This Work


2^n
n^100
n^4
n^3
n^2
Hardness
K-SAT
100-OV
4-OV
3-OV
APSP
Triangle Collection
ZWT
OV
3SUM
[BallRosenSabinVasudevan'17] + This Work
FOV^100
FOV4
FOV^3
FTC
FZWT
FOV
F3SUM
PoW
non-interactive via fiat shamir
worst-case
non-amortizable
average-case
Proof of Work

Related Work

[Goldreich Rothblum '18]

Fine-grained
worst-case to average-case reductions
from/to (!) counting t-Cliques,

and corresponding PoWs.

Informal Mad-Lib Theorem

Assuming Orthogonal Vectors Conjecture / WORST-CASE CONJECTURE FROM FINE-GRAINED COMPLEXITY,

then the "recipe" (to be alluded to)

yields a Proof of Work Scheme

requiring quadratic / CONJECTURED HARDNESS work.

THE RECIPE

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction
- direct sum theorem: non-amortizable

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction
- direct sum theorem: non-amortizable

2 Efficiently delegate evaluation of FOV to the prover.

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction
- direct sum theorem: non-amortizable

2 Efficiently delegate evaluation of FOV to the prover.

- can check Prover evaluated FOV correctly

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction
- direct sum theorem: non-amortizable

2 Efficiently delegate evaluation of FOV to the prover.

- can check Prover evaluated FOV correctly
- does this too fast?

The Recipe

1 Represent Orthogonal Vectors as a low degree polynomial, FOV.

- fine-grained worst-case to average-case reduction breaks!
- direct sum theorem: non-amortizable

2 Efficiently delegate evaluation of FOV to the prover.

- can check Prover evaluated FOV correctly
- does this too fast?

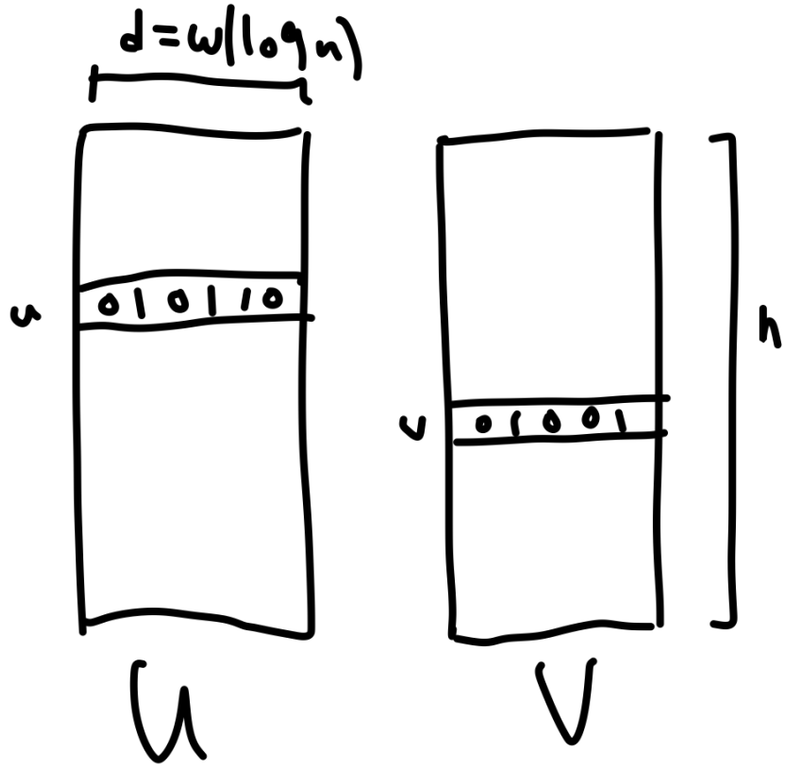
Representing OV as Polynomial

Representing OV as Polynomial

$$(u,v) \in OV \iff F_{OV}(u,v) \neq 0$$

Representing OV as Polynomial

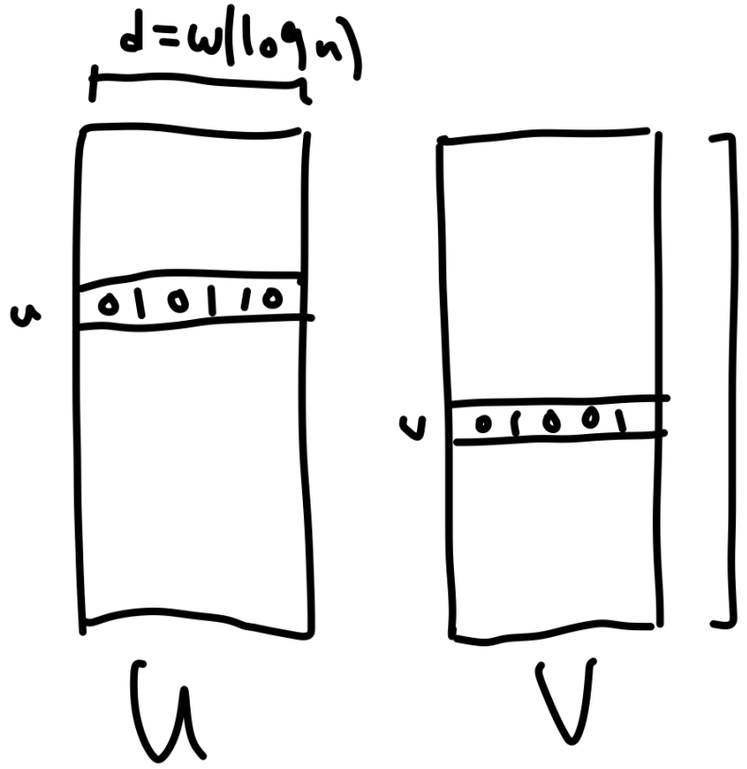
$$(U,V) \in OV \iff FOV(U,V) \neq 0$$



$$\exists \vec{u} \in U, \vec{v} \in V : \sum_{i=1}^{d} \vec{u}_i \vec{v}_i = 0?$$

Representing OV as Polynomial

$$(U,V) \in OV \iff FOV(U,V) \neq 0$$



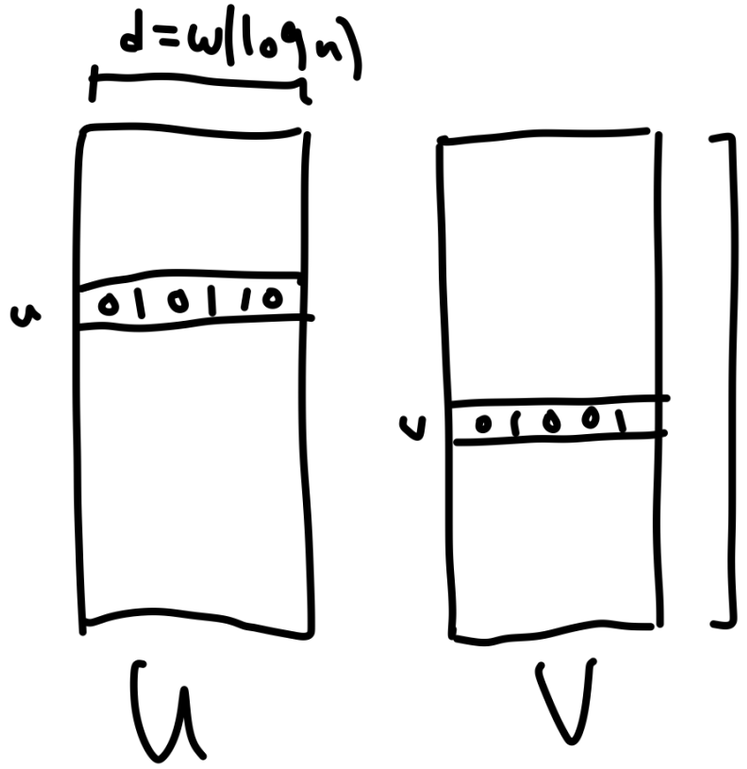
$$\exists \vec{u} \in U, \vec{v} \in V : \sum_{i=1}^{d} \vec{u}_i \vec{v}_i = 0?$$

$$\prod_{i=1}^{d} (1 - \vec{u}_i \vec{v}_i) \neq 0 \iff \langle \vec{u}, \vec{v} \rangle = 0$$

$(=1)$

Representing OV as Polynomial

$$(U,V) \in OV \iff FOV(U,V) \neq 0$$



$$\exists \vec{u} \in U, \vec{v} \in V : \sum_{i=1}^{d} \vec{u}_i \vec{v}_i = 0?$$

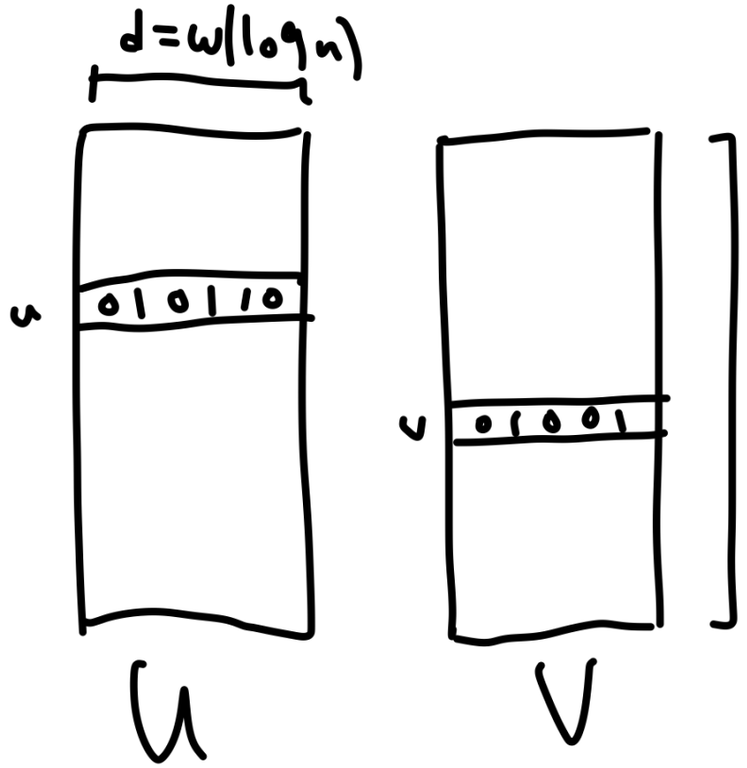
$$\prod_{i=1}^{d} (1 - \vec{u}_i \vec{v}_i) \neq 0 \iff \langle \vec{u}, \vec{v} \rangle = 0$$

$$(=1)$$

$$\sum_{\vec{u} \in U} \sum_{\vec{v} \in V} \prod_{i=1}^{d} (1 - \vec{u}_i \vec{v}_i) \neq 0 \iff (U,V) \in OV$$

Representing OV as Polynomial

$$(U,V) \in OV \iff FOV(U,V) \neq 0$$



$$\exists \vec{u} \in U, \vec{v} \in V : \sum_{i=1}^{d} \vec{u}_i \vec{v}_i = 0 ?$$

$$\prod_{i=1}^{d} (1 - \vec{u}_i \vec{v}_i) \neq 0 \iff \langle \vec{u}, \vec{v} \rangle = 0$$

$$(= 1)$$

$$\sum_{\vec{u} \in U} \sum_{\vec{v} \in V} \prod_{i=1}^{d} (1 - \vec{u}_i \vec{v}_i) \neq 0 \iff (U,V) \in OV$$

$$FOV : \mathbb{F}_p^{2nd} \to \mathbb{F}_p \qquad (p > n^2,\ \deg(FOV) = 2d)$$

Why Low Degree Poly?

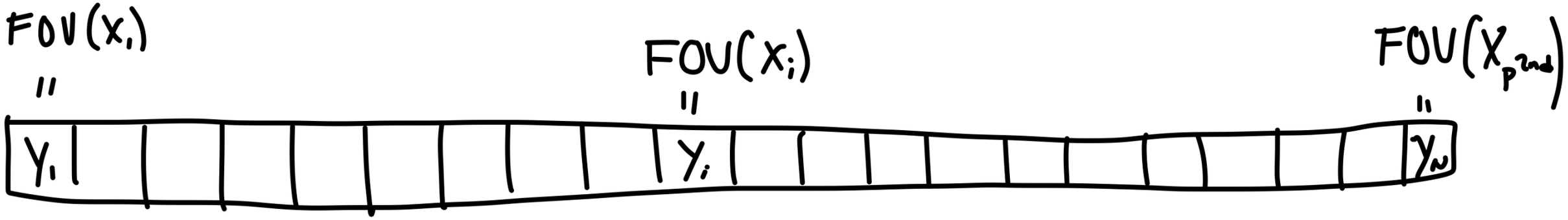
Why Low Degree Poly?

Fine-Grained Worst-case to Average-case Reduction [BRSV17]

Why Low Degree Poly?

Fine-Grained Worst-case to Average-case Reduction [BRSV17]

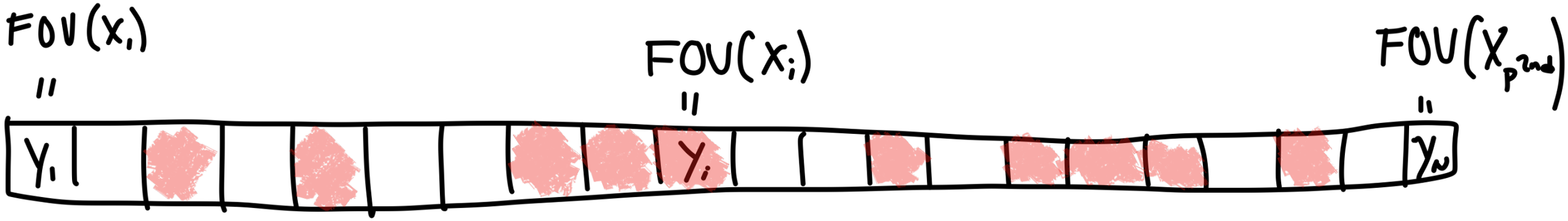
"Truth Table" of FOV = (Reed Muller) Codeword



Why Low Degree Poly?

Fine-Grained Worst-case to Average-case Reduction [BRSV17]

"Truth Table" of FOV = (Reed Muller) Codeword

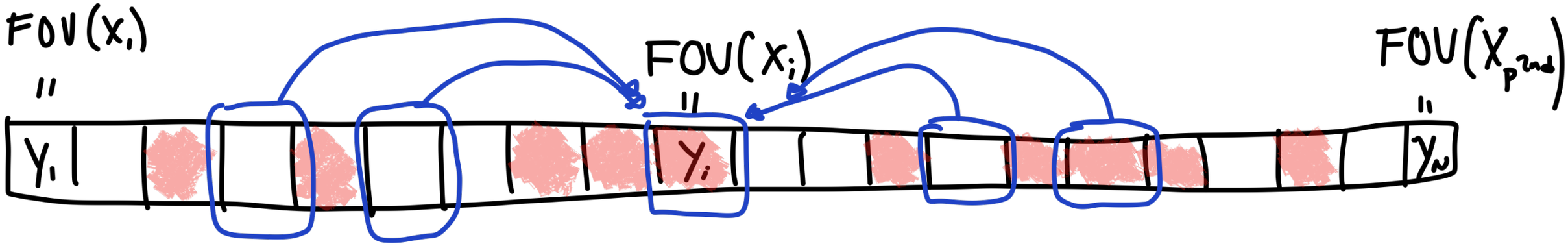


A wrong on "small" fraction of X_i's

Why Low Degree Poly?

Fine-Grained Worst-case to Average-case Reduction [BRSV17]

"Truth Table" of FOV = (Reed Muller) Codeword



A wrong on "small" fraction of X_i's

↓ (local reconstruction)

A' correct on <u>all</u> X_i's (w.h.p.)

FOV is Hard on Average

[Ball Rosen Sabin Vasudevan '17]

The Orthogonal Vectors Conjecture implies (or SETH)

any algorithm running in time $n^{2-\varepsilon}$ $(\varepsilon > 0)$

is correct (wrt FOV) on a uniform instance $X \in \mathbb{F}_p^{2nd}$

with probability $< 1/n^{o(1)}$

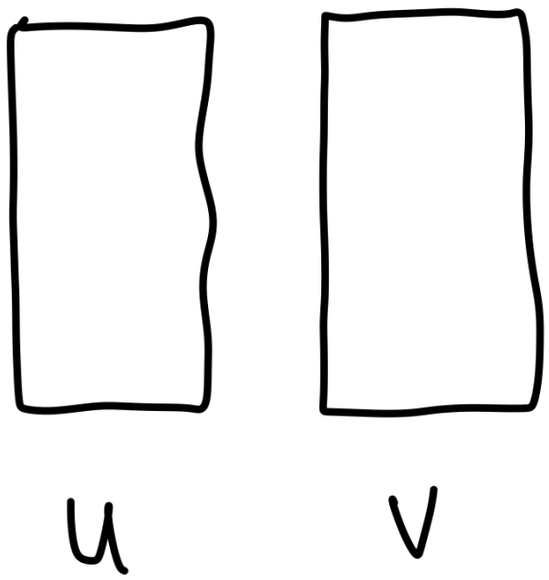
FOV is Non-Amortizable

FOV is Non-Amortizable

Key Idea: Downward Self-Reducibility

FOV is Non-Amortizable

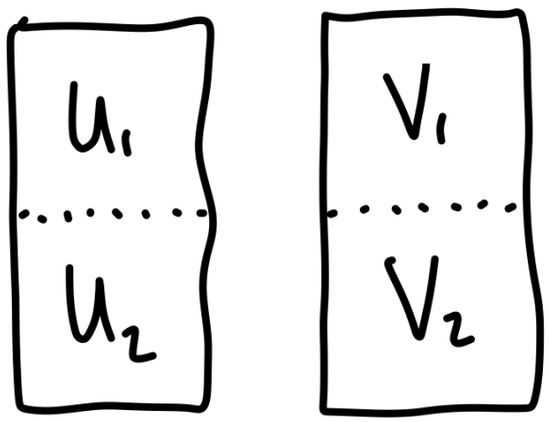
Key Idea: Downward Self-Reducibility



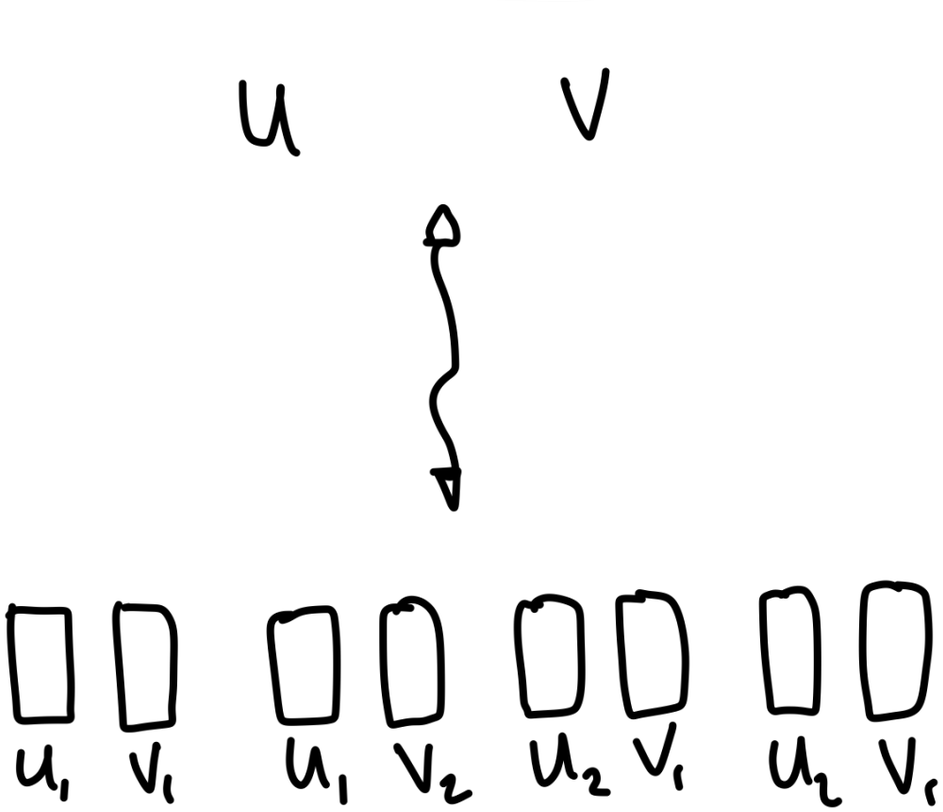
FOV(u,v)

FOV is Non-Amortizable

Key Idea: Downward Self-Reducibility

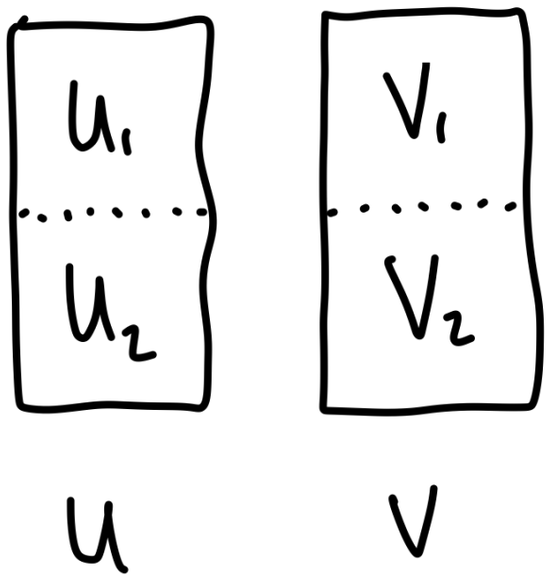


FOV(U,V)

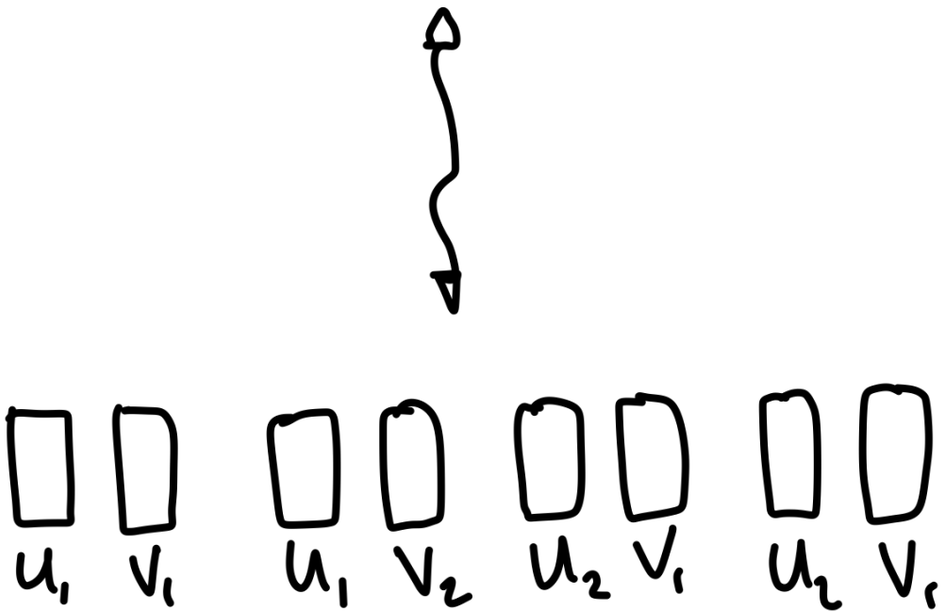


FOV is Non-Amortizable

Key Idea: Downward Self-Reducibility



$$FOV(U,V) = FOV(U_1, V_1) + FOV(U_1, V_2) + FOV(U_2, V_1) + FOV(U_2, V_2)$$



FOV is Non-Amortizable

The Orthogonal Vectors Conjecture implies
(or SETH)

any algorithm running in time $m(n) \cdot n^{2-\varepsilon}$

is correct (wrt FOV) on $m(n)$ uniform instances $X \in \mathbb{F}_p^{2nd}$

with probability $< 1/n^{o(1)}$

$(m(n) = \text{poly}(n)\ ,\ \varepsilon > 0)$

FOV is Non-Amortizable

The Orthogonal Vectors Conjecture implies
(or SETH)

any algorithm running in time $m(n) \cdot n^{2-\varepsilon}$

is correct (wrt FOV) on $m(n)$ uniform instances $X \in \mathbb{F}_p^{2nd}$

with probability $< 1/n^{o(1)}$

$(m(n) = \text{poly}(n)\ ,\ \varepsilon > 0)$

NOTE: Batch <u>univariate</u> polynomial evaluation is easy!

[Fiduccia'72, Horowitz'72]

The Recipe

1 ✓ Represent Orthogonal Vectors as a low degree polynomial, FOV.

= fine-grained worst-case to average-case reduction ✓
direct sum theorem: non-amortizable

breaks!

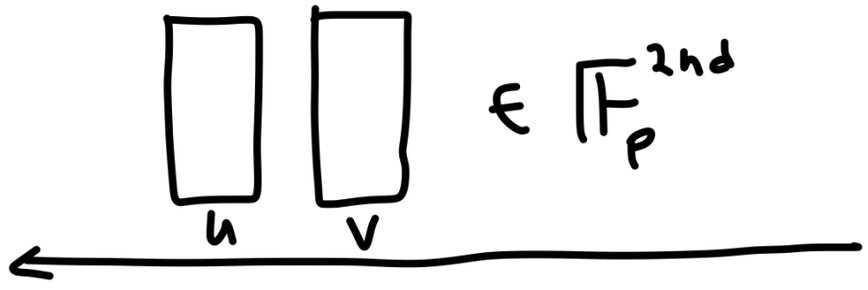
2 Efficiently delegate evaluation of FOV to the prover.

- can check Prover evaluated FOV correctly
- does this too fast?

DELEGATING FOV

(SUM CHECK)

PROVER

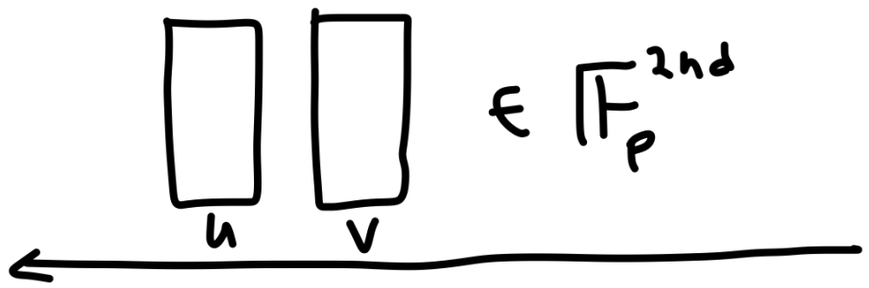


VERIFIER

DELEGATING FOV

(SUM CHECK)

PROVER



VERIFIER

Compute:

(a) $FOU(u, v) = y$

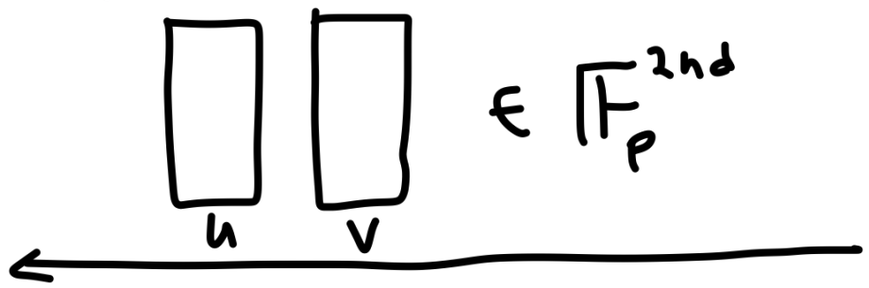
(b) coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$

DELEGATING FOV

(SUM CHECK)

PROVER



VERIFIER

Compute:

ⓐ $FOV(u,v)=y$

ⓑ coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$ →

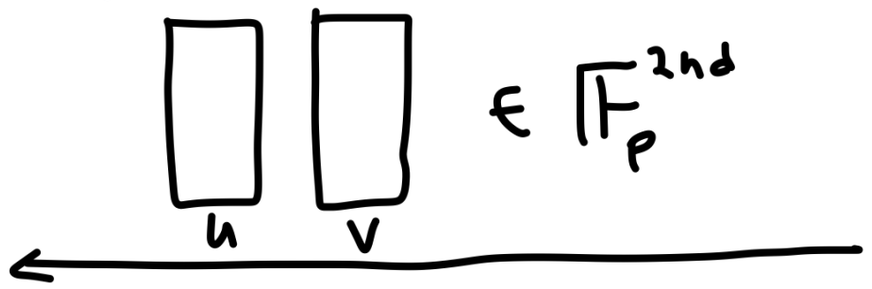
Check:

① what prover sent is (the unique) $G_{uv}(\cdot)$

② $y = FOV(u,v)$

DELEGATING FOV (SUM CHECK)

PROVER



VERIFIER

Compute:

(a) $FOV(u,v) = y$

(b) coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$ ⟶

$G_{uv}(\cdot)$ ↳ univariate poly of deg $\tilde{O}(n)$:

Check:

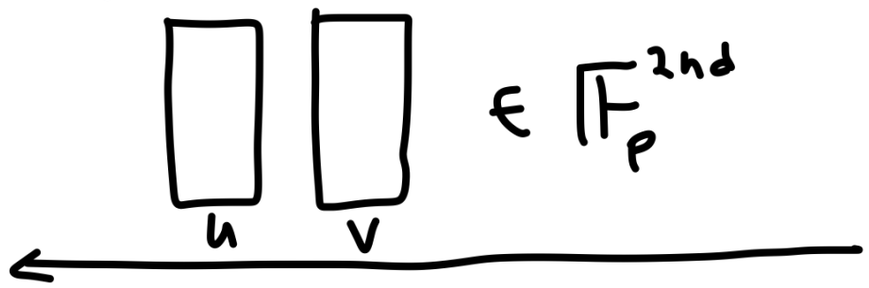
(1) what prover sent is (the unique) $G_{uv}(\cdot)$

(2) $y = FOV(u,v)$

DELEGATING FOV

(SUM CHECK)

PROVER



VERIFIER

Compute:

ⓐ $FOV(u,v)=y$

ⓑ coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$ →

Check:

① what prover sent is (the unique) $G_{uv}(\cdot)$

② $y = FOV(u,v)$

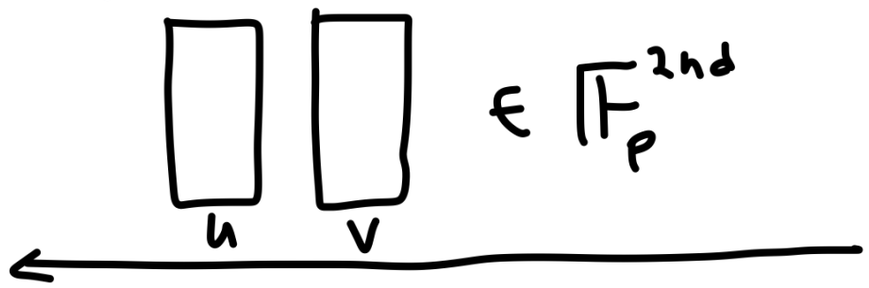
univariate poly of deg $\tilde{O}(n)$:

- Can compute ⓑ in time $\tilde{O}(n^2)$

DELEGATING FOV

(SUM CHECK)

PROVER



VERIFIER

Compute:

ⓐ $FOV(u,v)=y$

ⓑ coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$ →

univariate poly of deg $\tilde{O}(n)$:

- can compute ⓑ in time $\tilde{O}(n^2)$
- can check ① in time $\tilde{O}(n)$

Check:

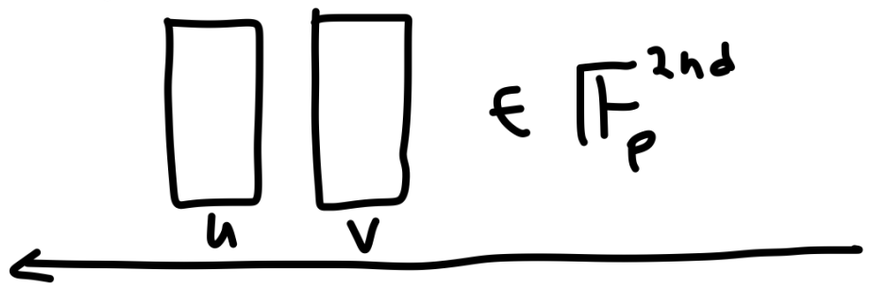
① what prover sent is (the unique) $G_{uv}(\cdot)$

② $y = FOV(u,v)$

DELEGATING FOV

(SUM CHECK)

PROVER



VERIFIER

Compute:

ⓐ $FOV(u,v)=y$

ⓑ coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$ ⟶

Check:

① what prover sent is (the unique) $G_{uv}(\cdot)$

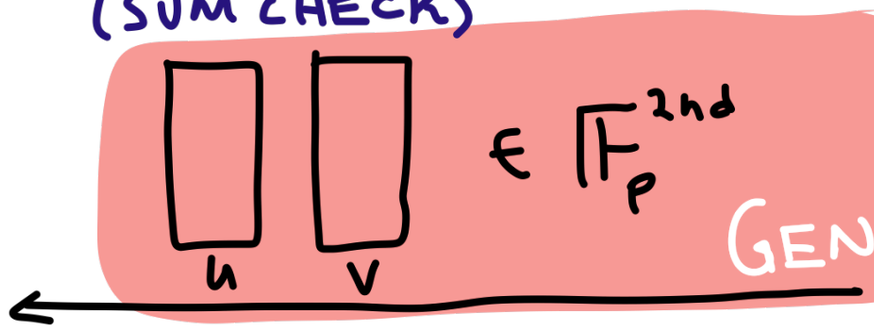
② $y = FOV(u,v)$

univariate poly of deg $\tilde{O}(n)$:

- can compute ⓑ in time $\tilde{O}(n^2)$
- can check ① in time $\tilde{O}(n)$
- given correct $G_{uv}(\cdot)$, can check ② in time $\tilde{O}(n)$

DELEGATING FOV

(SUM CHECK)



PROVER

PROVE

Compute:

ⓐ $FOV(u,v)=y$

ⓑ coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$

VERIFIER

VERIFY

Check:

① what prover sent is (the unique) $G_{uv}(\cdot)$

② $y = FOV(u,v)$

univariate poly of deg $\tilde{O}(n)$:

- can compute ⓑ in time $\tilde{O}(n^2)$
- can check ① in time $\tilde{O}(n)$
- given correct $G_{uv}(\cdot)$, can check ② in time $\tilde{O}(n)$

Proof of Work Scheme

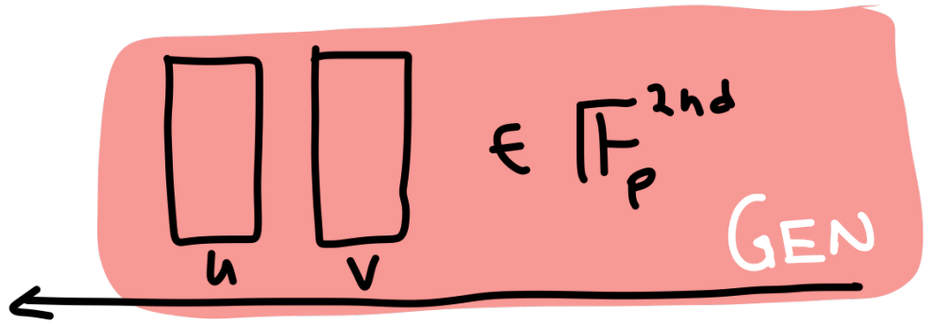
PROVER

PROVE

Compute:

(a) $FOU(u,v)=y$

(b) coefficients of $G_{uv}(\cdot)$



VERIFIER

$y, G_{uv}(\cdot)$

VERIFY

Check:

(1) what prover sent is (the unique) $G_{uv}(\cdot)$

(2) $y = FOV(u,v)$

Efficiency ✓

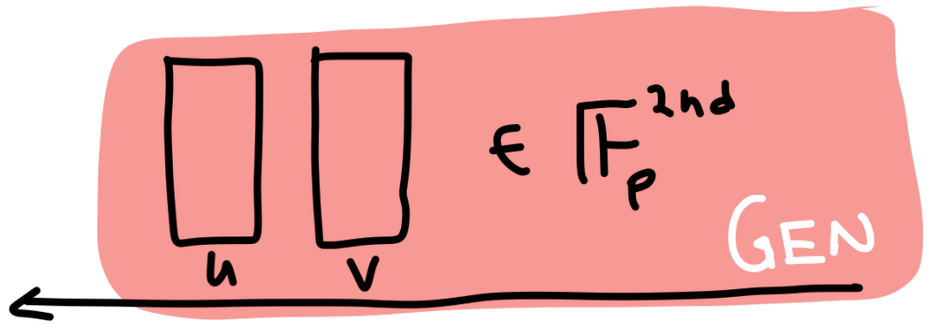
Completeness ✓

Hardness ✓

Proof of Work Scheme

PROVER

VERIFIER



PROVE

Compute:

(a) $FOU(u,v)=y$

(b) coefficients of $G_{uv}(\cdot)$

$y, G_{uv}(\cdot)$

not needed

VERIFY

Check:

(1) what prover sent is (the unique) $G_{uv}(\cdot)$

(2) $y = FOV(u,v)$

Efficiency ✓

Completeness ✓

Hardness ✓

What We Just Saw

Assuming Orthogonal Vectors Conjecture / worst-case conjecture from fine-grained complexity,

then the "recipe" (described earlier) yields a Proof of Work scheme requiring quadratic / conjectured hardness work,

WHAT WE JUST SAW

Assuming Orthogonal Vectors Conjecture / WORST-CASE CONJECTURE FROM FINE-GRAINED COMPLEXITY,

then the "recipe" (described earlier) yields a Proof of Work Scheme requiring quadratic / CONJECTURED HARDNESS work,

with: (a) 'unique' proofs

WHAT WE JUST SAW

Assuming $\frac{\text{Orthogonal Vectors Conjecture}}{\text{WORST-CASE CONJECTURE FROM FINE-GRAINED COMPLEXITY}}$,

then the "recipe" (described earlier) yields a Proof of Work Scheme requiring $\frac{\text{quadratic}}{\text{CONJECTURED HARDNESS}}$ work,

with: (a) 'unique' proofs

(b) 'simple' algebraic structure

WHAT WE JUST SAW

Assuming $\frac{\text{Orthogonal Vectors Conjecture}}{\text{WORST-CASE CONJECTURE FROM FINE-GRAINED COMPLEXITY}}$,

then the "recipe" (described earlier) yields a Proof of Work Scheme requiring $\frac{\text{quadratic}}{\text{CONJECTURED HARDNESS}}$ work,

with: (a) 'unique' proofs

(b) 'simple' algebraic structure

CAVEAT: "harder" Polynomials yield interactive Protocols.

Exploiting Algebraic Structure

Exploiting Algebraic Structure in Zero-Knowledge Proofs of Work

Exploiting Algebraic Structure

Zero-Knowledge Proofs of Work

verifier learns "nothing" from prover
(beyond that he performed work)

Exploiting Algebraic Structure

Zero-Knowledge Proofs of Work

verifier learns "nothing" from prover
(beyond that he performed work)

↑ Can efficiently simulate interactions with (honest) verifier

Exploiting Algebraic Structure

Zero-Knowledge Proofs of Work

verifier learns "nothing" from prover
(beyond that he performed work)

Can efficiently simulate interactions with (honest) verifier

in time $\tilde{O}(n)$

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{MV})
and ② compare to a value he can compute himself.

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{MV})
and ② compare to a value he can compute himself.

IDEA: assume DDH (ElGamal), (mimic DLOG equality)

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{UV})
and ② compare to a value he can compute himself.

IDEA: assume DDH (ElGamal), (mimic DLOG equality)

→ PROVER sends pk, homomorphic commitment to G_{UV}

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{UV})
and ② compare to a value he can compute himself.

↓

IDEA: assume DDH (ElGamal), (mimic DLOG equality)

→ Prover sends pk, homomorphic commitment to G_{UV}

① Homomorphically evaluate G_{UV}

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{UV})
and ② compare to a value he can compute himself.

IDEA: assume DDH (ElGamal), (mimic DLOG equality)

→ PROVER sends pk, homomorphic commitment to G_{UV}

① Homomorphically evaluate G_{UV}

② commit to the value

ZKPoW

PoW: all verifier needs to do is ① evaluate a polynomial (G_{UV})
and ② compare to a value he can compute himself.

↓

IDEA: assume DDH (ElGamal), (mimic DLOG equality)

→ Prover sends pk, homomorphic commitment to G_{UV}

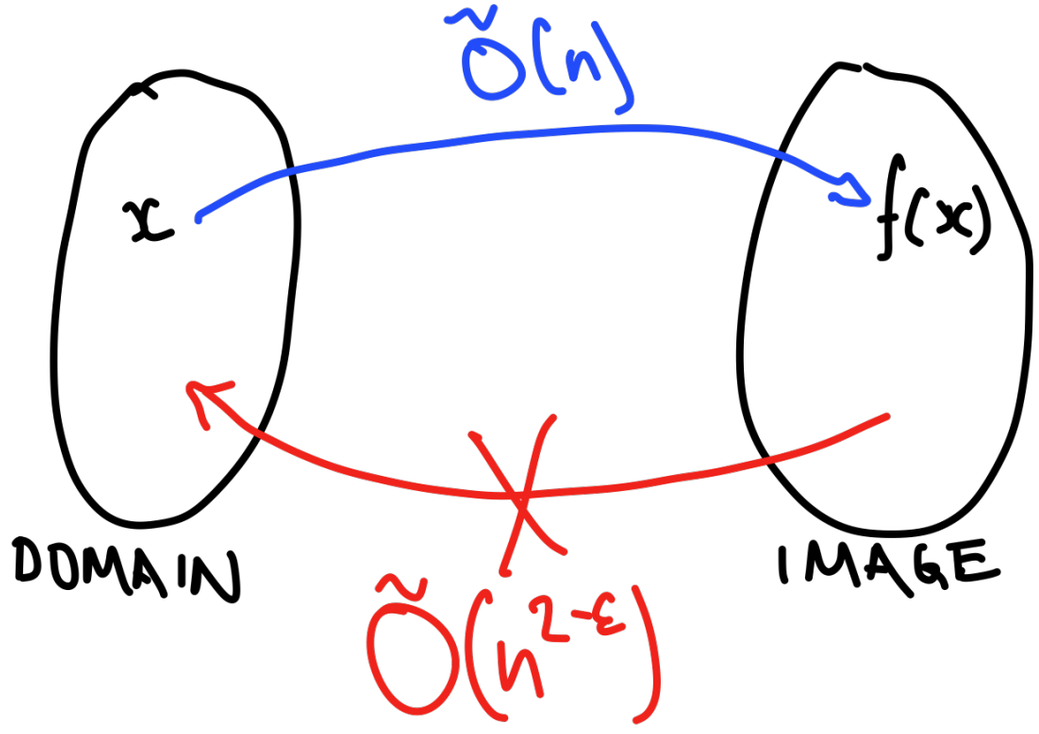
① Homomorphically evaluate G_{UV}

② commit to the value

→ Test equality of ① & ②

OPEN

Moderately Hard One-way Functions (from similar assumptions)



THANKS!

[Ball Rosen Sabin Vasudevan '17] + This Work

2^n

n^{100}

n^4

n^3

n^2

Hardness

k-SAT

100-OV

4-OV

3-OV

OV

APSP

3SUM

Triangle Collection

ZWT

$\#\text{OV}^{100}$

$\#\text{OV}^4$

$\#\text{OV}^3$

#OV

#ZWT

#TC

#ZWT

#OV

#3SUM

PoW

PoW

PoW

PoW

PoW

PoW

PoW

non-interactive via fiat shamir

worst-case

non-amortizable

average-case

Proof of Work