A Simple and Efficient Framework of Proof Systems for NP

Yuyu Wang[1], Chuanjie Su[1], Jiaxin pan[2], Yu Chen[3]

1. University of Electronic Science and Technology of China
2. NTNU - Norwegian University of Science and Technology
3. Shandong University

Proof systems

Non-interactive zero-knowledge proof (NIZK)

Non-interactive batch argument (BARG)

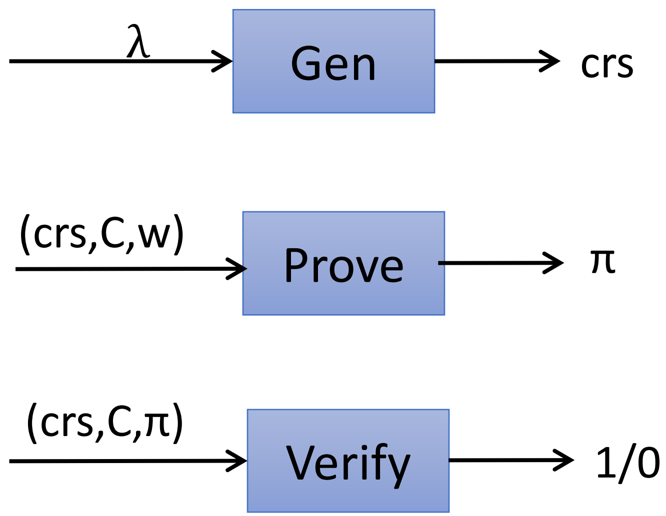
Proof systems

Non-interactive zero-knowledge proof (NIZK)

Non-interactive batch argument (BARG)

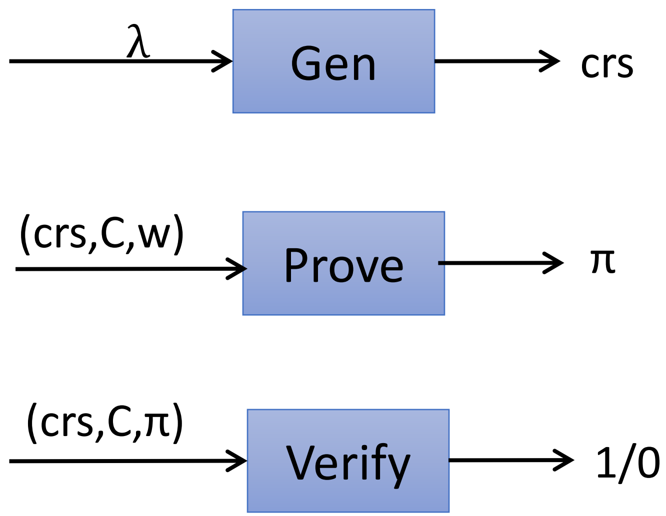
Definition of NIZK for NP

$$L^{CSAT} = \{C \mid \exists w: C(w) = 1\}$$



Definition of NIZK for NP

$$L^{CSAT} = \{C \mid \exists w : C(w) = 1\}$$



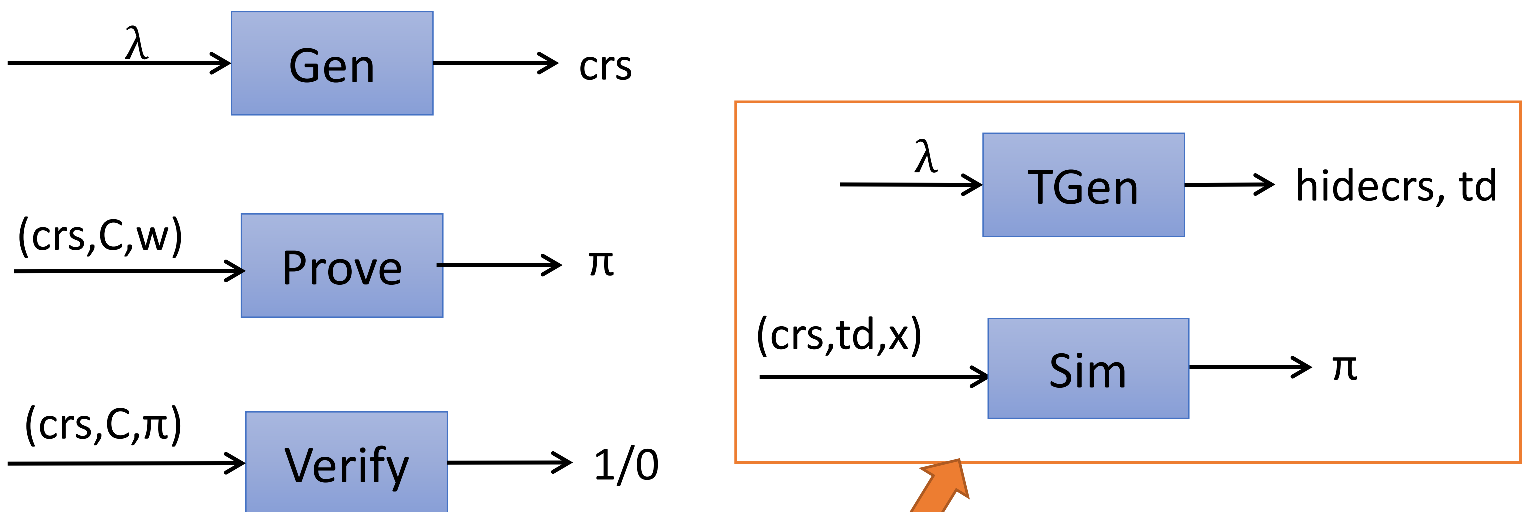
Completeness: honest proofs must pass the verification.

Soundness: difficult to find a valid proof for any invalid statement.

Zero-knowledge: π reveals no additional information on w except for the statement.

Definition of NIZK for NP

$$L^{CSAT} = \{C \mid \exists w: C(w) = 1\}$$



Completeness: honest proofs must pass the verification.

Soundness: difficult to find a valid proof for any invalid statement.

Zero-knowledge: π reveals no additional information on w except for the statement.

Existing NIZK for NP

Assumptions:

- Quadratic residuosity, trapdoor permutation [BFM88,FLS99]
- DLIN, subgroup decision (in pairings) [GOS06]
- LWE [PS19]
- Non-falsifiable assumptions [Groth12,Lipmaa12,GGPR13]
- CDH*+DLIN ([KKNY19,KKNY20])

Existing NIZK for NP

Assumptions:

- Quadratic residuosity, trapdoor permutation [BFM88,FLS99]
- DLIN, subgroup decision (in pairings) [GOS06]
- LWE [PS19]
- Non-falsifiable assumptions [Gr
- CDH*+DLIN ([KKNY19,KKNY20])

State-of-the-art in the pairings

Existing NIZK for NP

Assumptions:

- Quadratic residuosity, trapdoor permutation [BFM88,FLS99]
- DLIN, subgroup decision (in pairings) [GOS06]
- LWE [PS19]
- Non-falsifiable assumptions [Groth12,Lipn GGPR13]
- CDH*+DLIN ([KKNY19,KKNY20])

Is it possible to improve the efficiency of GOS-NIZK without any trade-off?

Our Results

Pairing-based NIZK for NP with shorter proofs and less proving and verification cost than GOS-NIZK.

Our Results

Pairing-based NIZK for NP with shorter proofs and less proving and verification cost than GOS-NIZK.

We consider Type-3 pairings, since it is the most efficient one among all types of pairings.

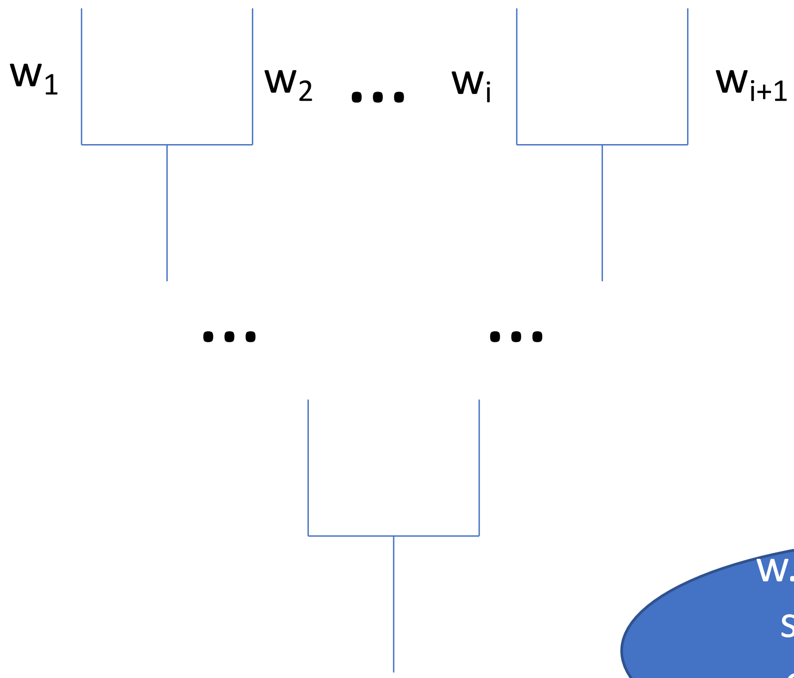
Our Results

Pairing-based NIZK for NP with shorter proofs and less proving and verification cost than GOS-NIZK.

Assumption: MDDH assumptions.

NIZK for NP [GOS06]

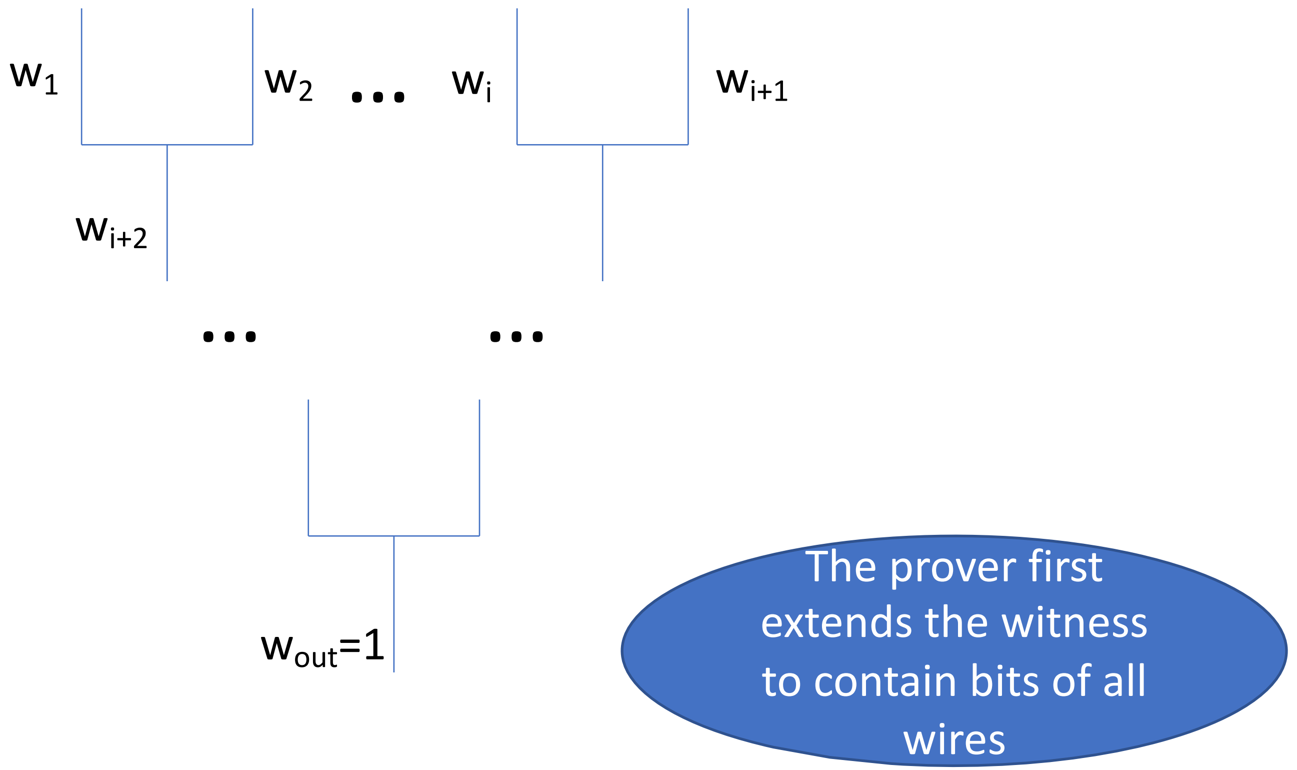
Prover:



w.l.o.g., we consider statement circuits consisting only of NAND gates

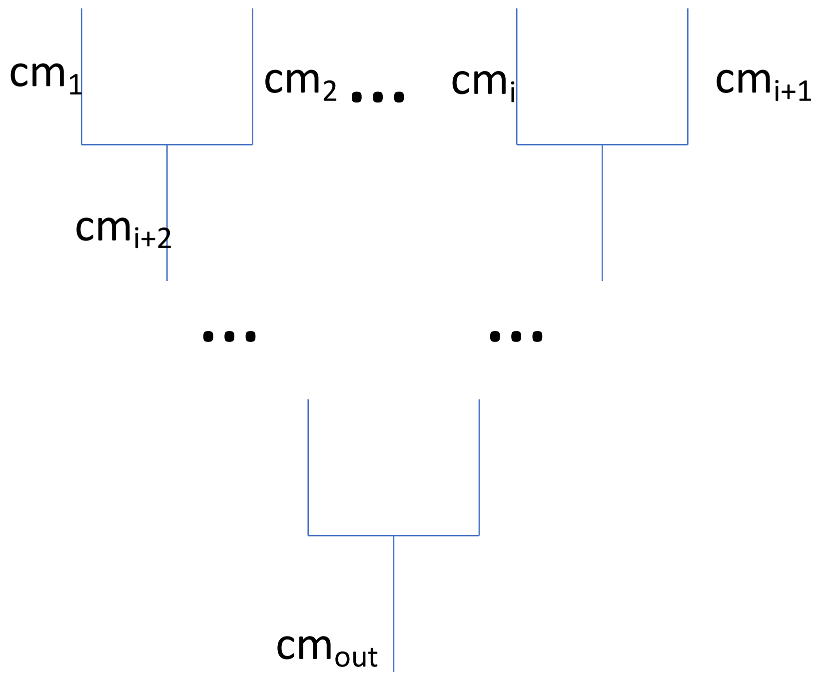
NIZK for NP [GOS06]

Prover:



NIZK for NP [GOS06]

Prover:

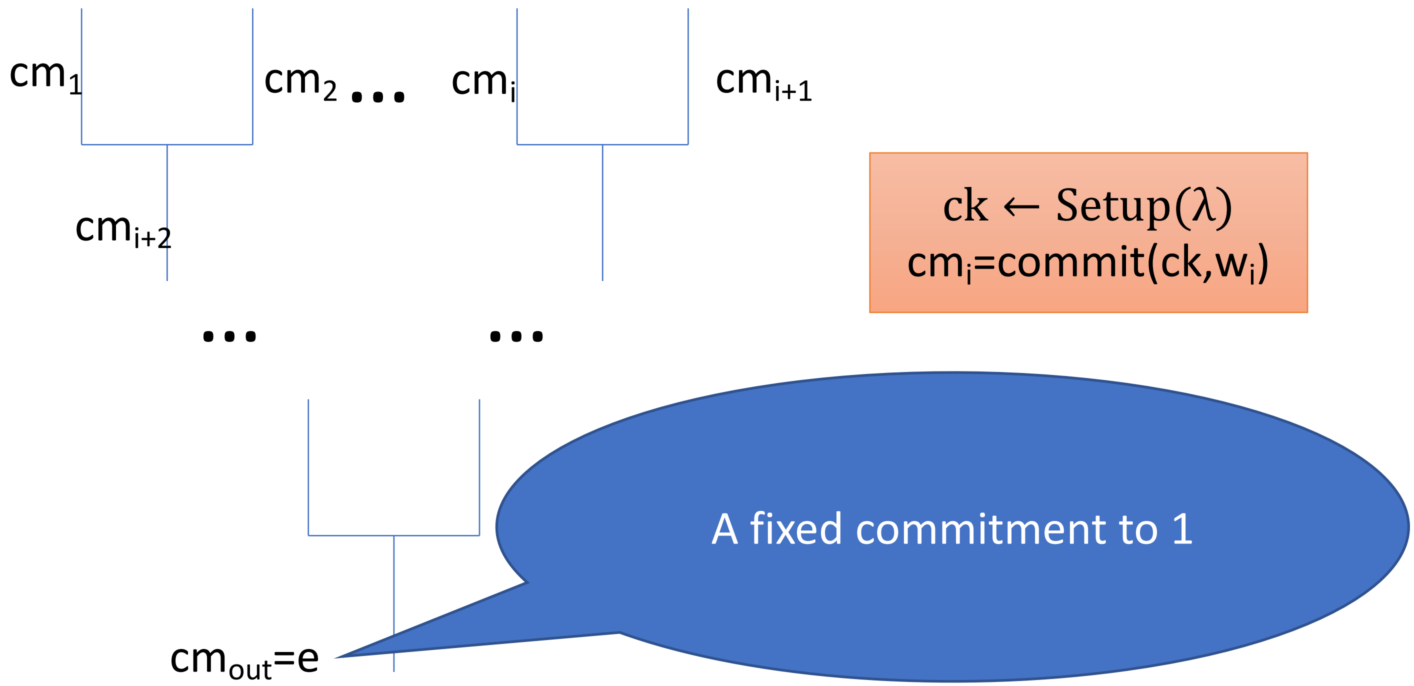


$ck \leftarrow \text{Setup}(\lambda)$
$cm_i = \text{commit}(ck, w_i)$

Additive homomorphic commitment

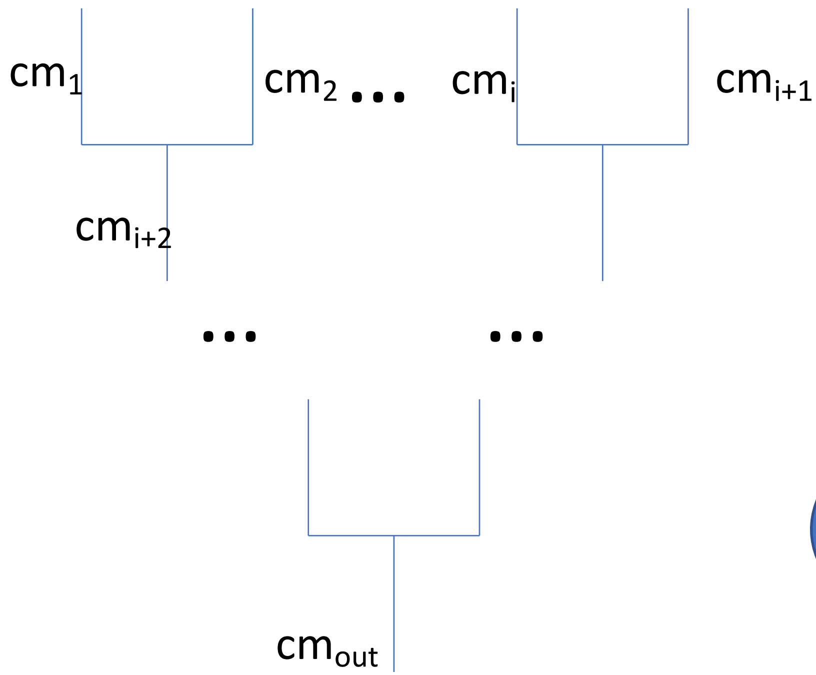
NIZK for NP [GOS06]

Prover:



NIZK for NP [GOS06]

Prover:

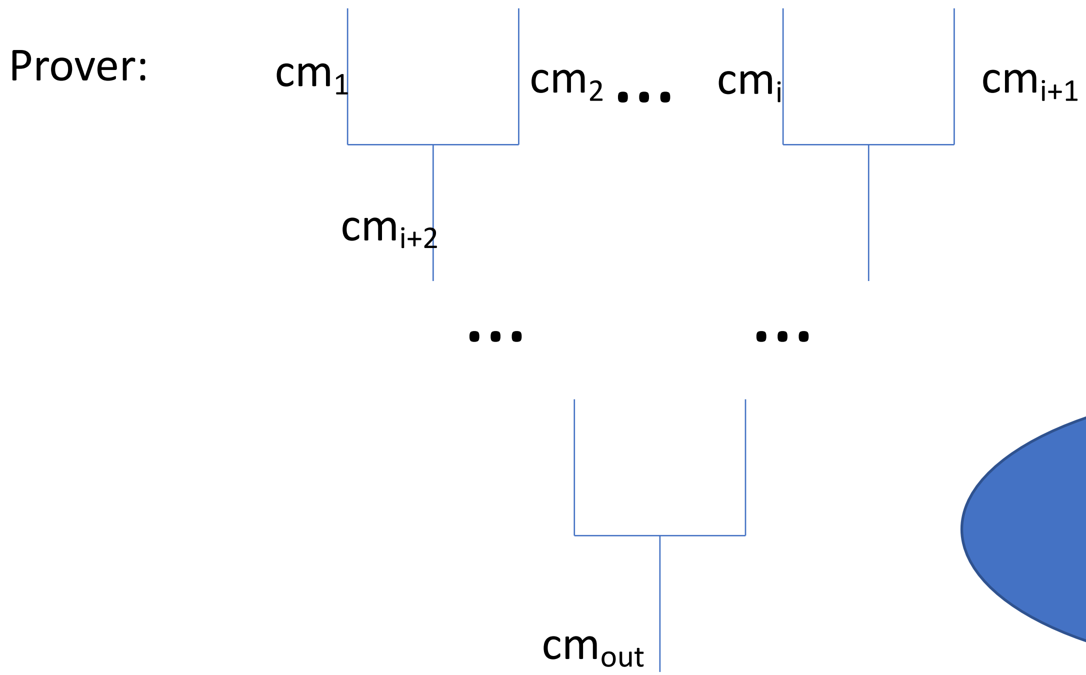


$ck \leftarrow \text{Setup}(\lambda)$

$cm_i = \text{commit}(ck, w_i)$

Hiding property

NIZK for NP [GOS06]

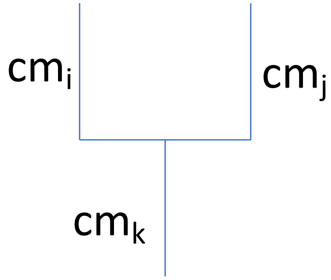


$\mathrm{ck} \leftarrow \mathrm{Setup}(\lambda)$
cm_i=commit(ck,w_i)

There is a trapdoor that can be used to extract the committed values

NIZK for NP [GOS06]

Prover:



NAND gate

The prover proves that the input/output commitments satisfy a relation supported by an OR-proof.

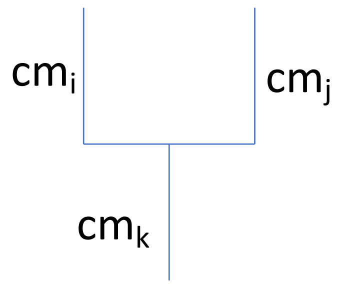
$cm_i + cm_j + cm_k - 2e$ commits to 0 or 1

and

cm_i , cm_j, cm_k commit to 0 or 1

NIZK for NP [GOS06]

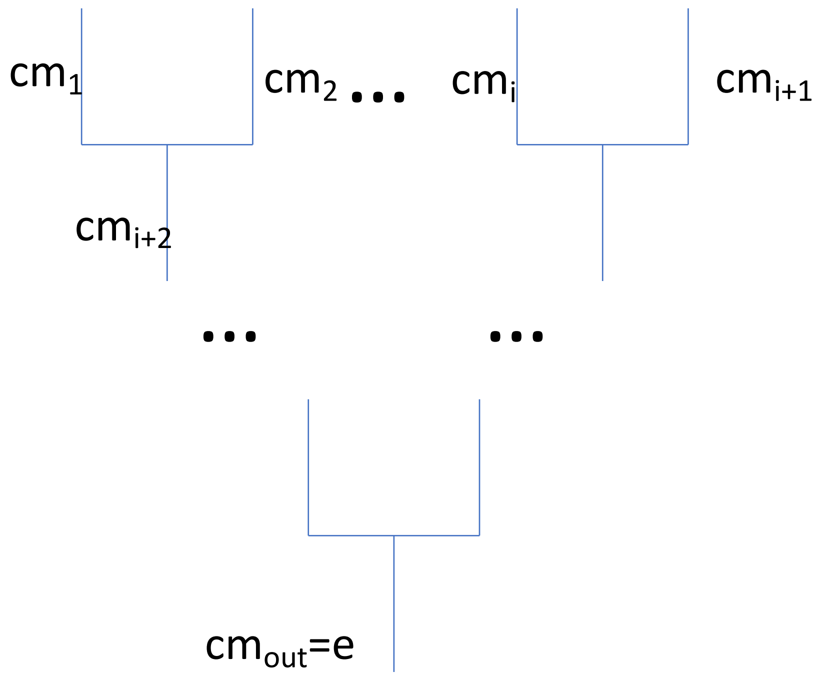
Verifier:



NAND gate

The verifier checks the validity of OR-proofs and whether the output commitment is e.

NIZK for NP [GOS06]



$ck \leftarrow Setup(\lambda)$
cm_i=commit(ck,w_i)

Zero-knowledge: hiding property of the commitment and the zero-knowledge of the underlying OR-proof.

NIZK for NP [GOS06]

Soundness:

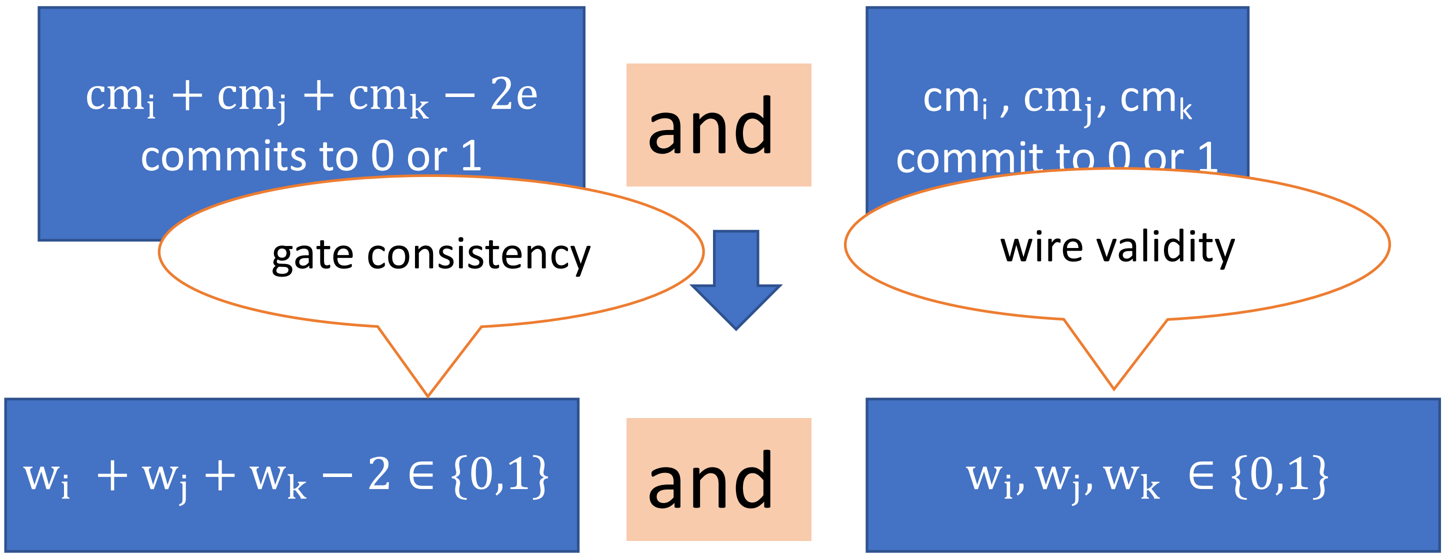
$\mathrm{cm_i} + \mathrm{cm_j} + \mathrm{cm_k} - 2\mathrm{e}$
commits to 0 or 1

and

$\mathrm{cm_i}$, $\mathrm{cm_j}$, $\mathrm{cm_k}$
commit to 0 or 1

NIZK for NP [GOS06]

Soundness:



NIZK for NP [GOS06]

Soundness:

$\mathrm{cm_i} + \mathrm{cm_j} + \mathrm{cm_k} - 2\mathrm{e}$
commits to 0 or 1

and

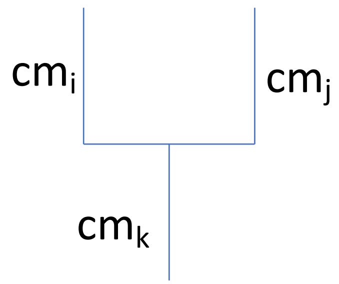
$\mathrm{cm_i}$, $\mathrm{cm_j}$, $\mathrm{cm_k}$
commit to 0 or 1

$\mathrm{w_i} \ + \mathrm{w_j} + \mathrm{w_k} - 2 \in \{0,1\}$

Then we can extract a
valid witness from any
valid proof.

Our Technique: Proving an Alternative Relation

Prover:



NAND gate

The prover proves that the commitments satisfy another relation supported by the OR-proof.

$\mathrm{e} - \mathrm{cm_i} - \mathrm{cm_k}$ commits to 0 and e-cm$_j$ commits to 0	or	$\mathrm{e} - \mathrm{cm_k}$ commits to 0 and $\mathrm{cm_j}$ commits to 0

Proving an Alternative Relation

$e - cm_i - cm_k$ commits to 0
and
e-cm_j commits to 0

or

$e - cm_k$ commits to 0
and
cm_j commits to 0

Cost is less if we adopt this relation

Proving an Alternative Relation

gate consistency is satisfied

$e - cm_i - cm_k$ commits to 0
and
e-cm_j commits to 0

or

$e - cm_k$ commits to 0
and
cm_j commits to 0

↓

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

Proving an Alternative Relation

$e - cm_i - cm_k$ commits to 0
and
e-cm_j commits to 0

or

e

cm_j commits to 0

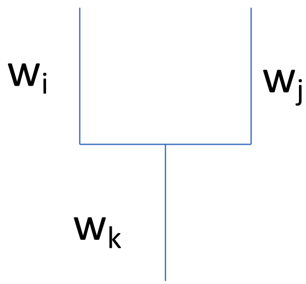
wire validity is unclear

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

Problems



NAND gate

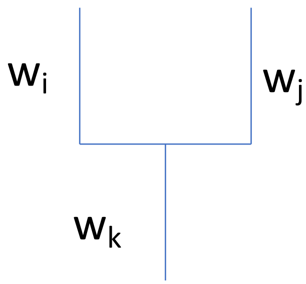
When w_j=1, w_i and w_k might be large numbers with the sum "happening to be" 1, e.g., $w_i + w_k = 5 + 9 \mod 13$

$$1 - w_i - w_k = 0$$
and
$$1 - w_j = 0$$

or

$$1 - w_k = 0$$
and
$$w_j = 0$$

Problems



NAND gate

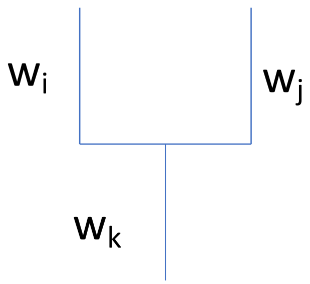
When w_j=0, w_i might be any large value

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

Problems



NAND gate

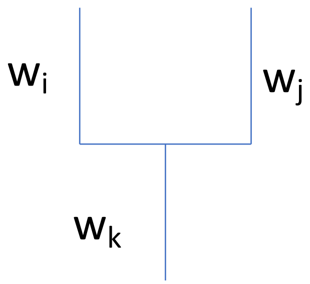
Additionally prove that the committed values are binary?

$$1 - w_i - w_k = 0$$
and
$$1 - w_j = 0$$

or

$$1 - w_k = 0$$
and
$$w_j = 0$$

Problems



NAND gate

Additionally prove that the committed values are binary?

Less efficient than GOS-NIZK

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

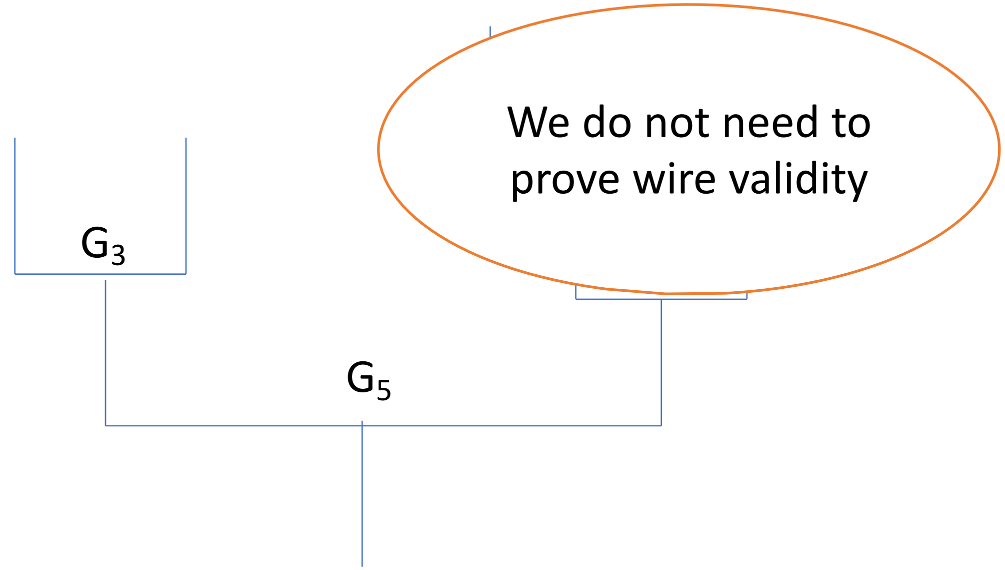
$1 - w_k = 0$
and
$w_j = 0$

New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

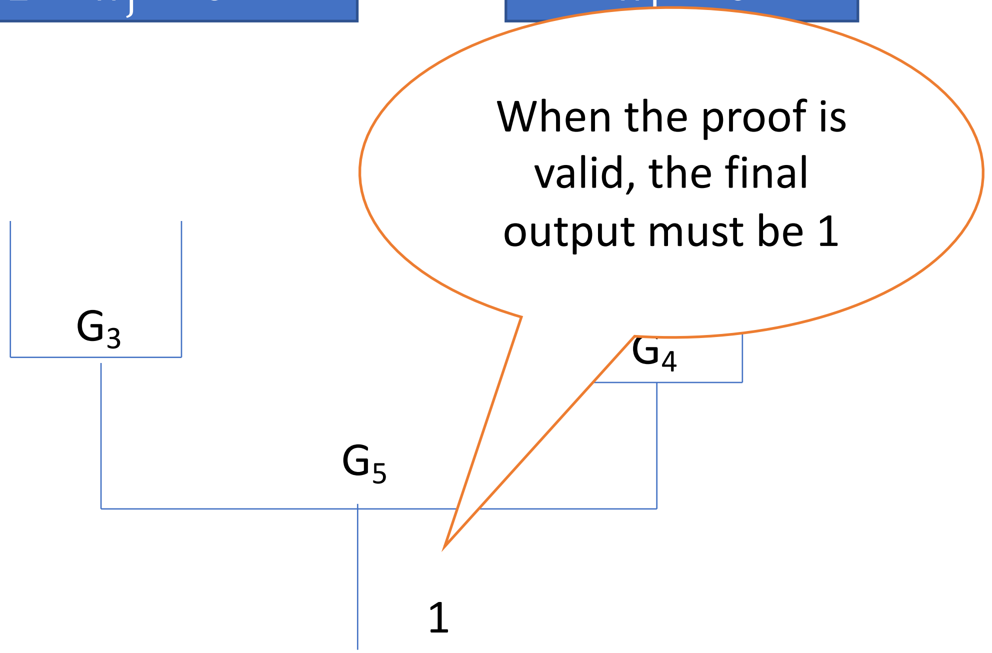


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_i = 0$

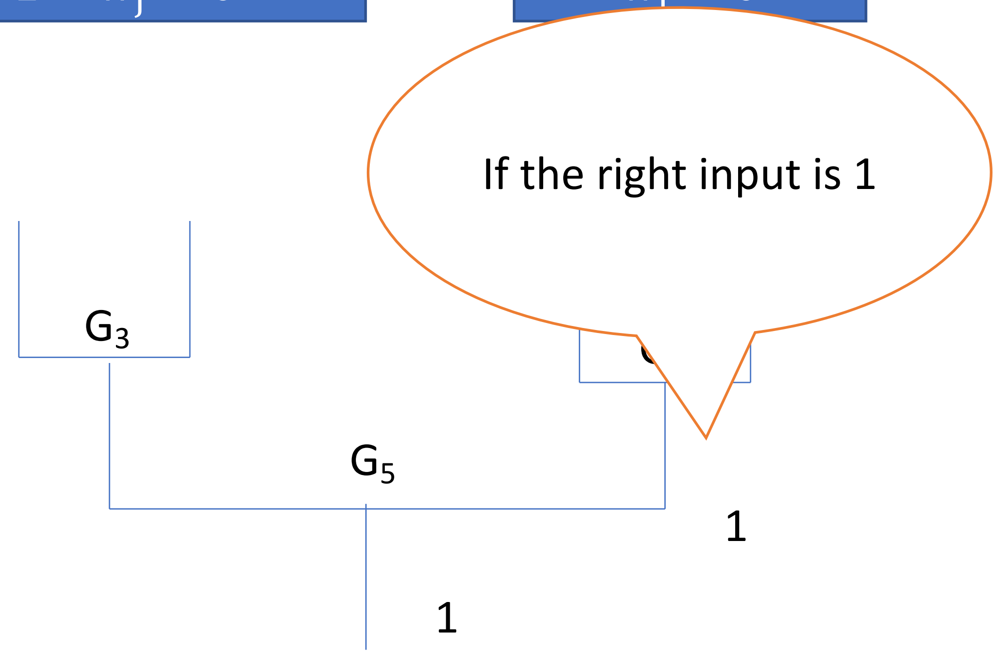


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

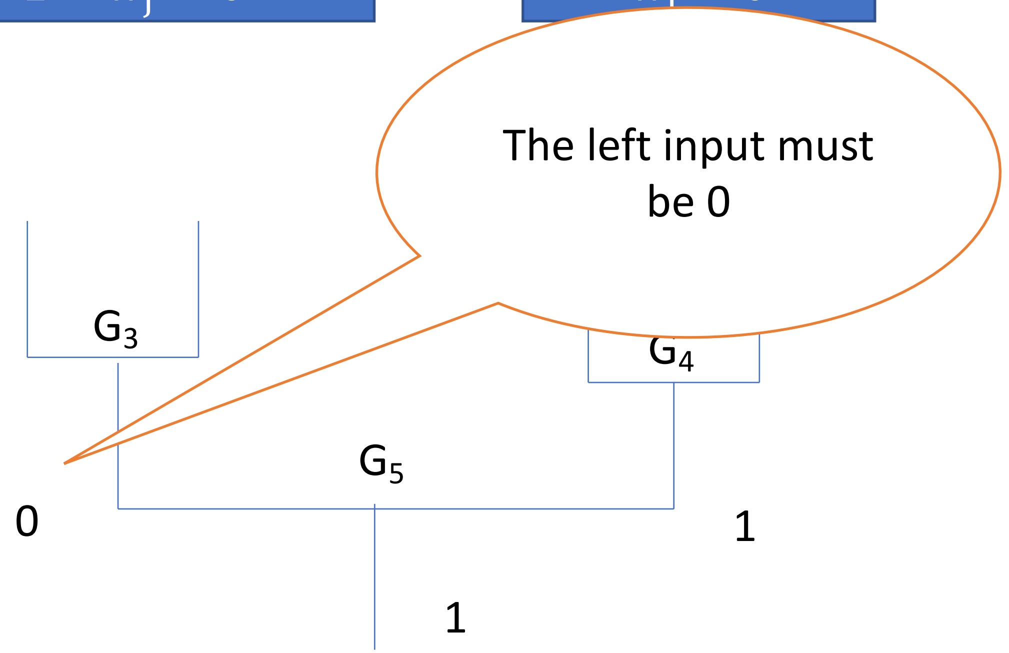


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

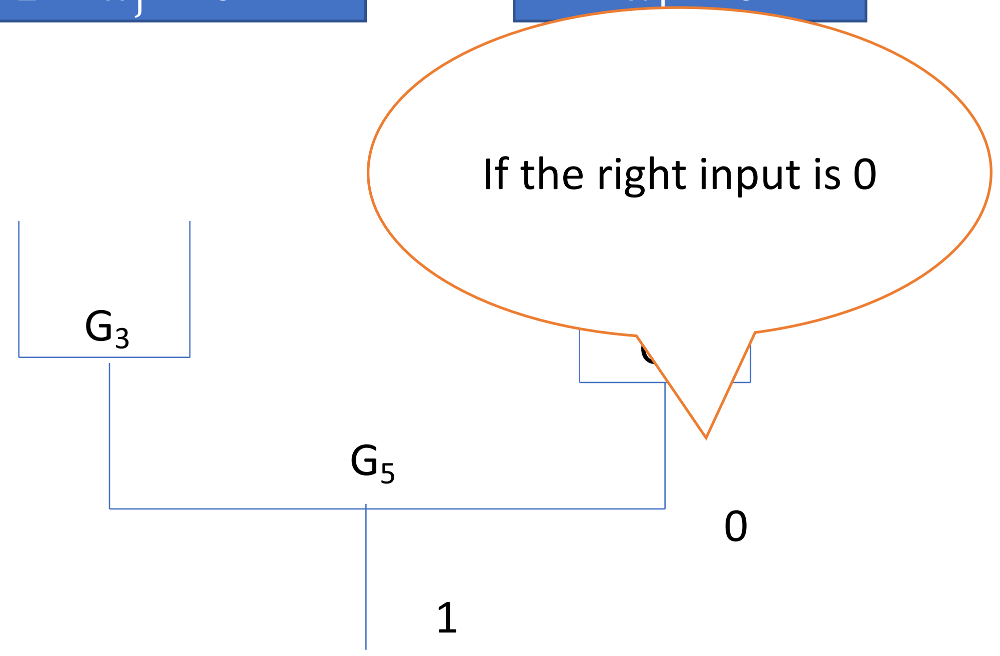


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

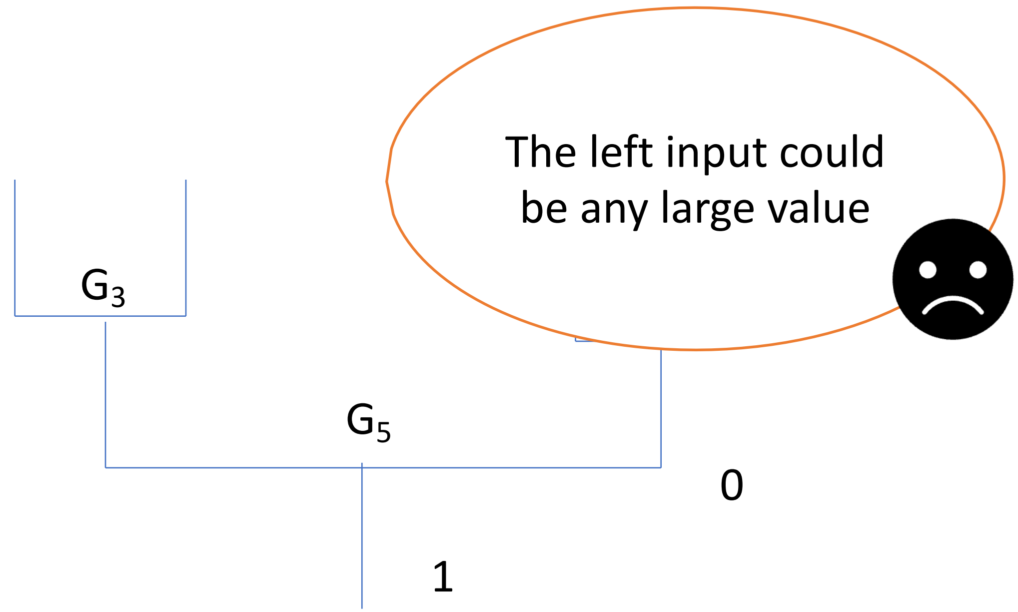


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

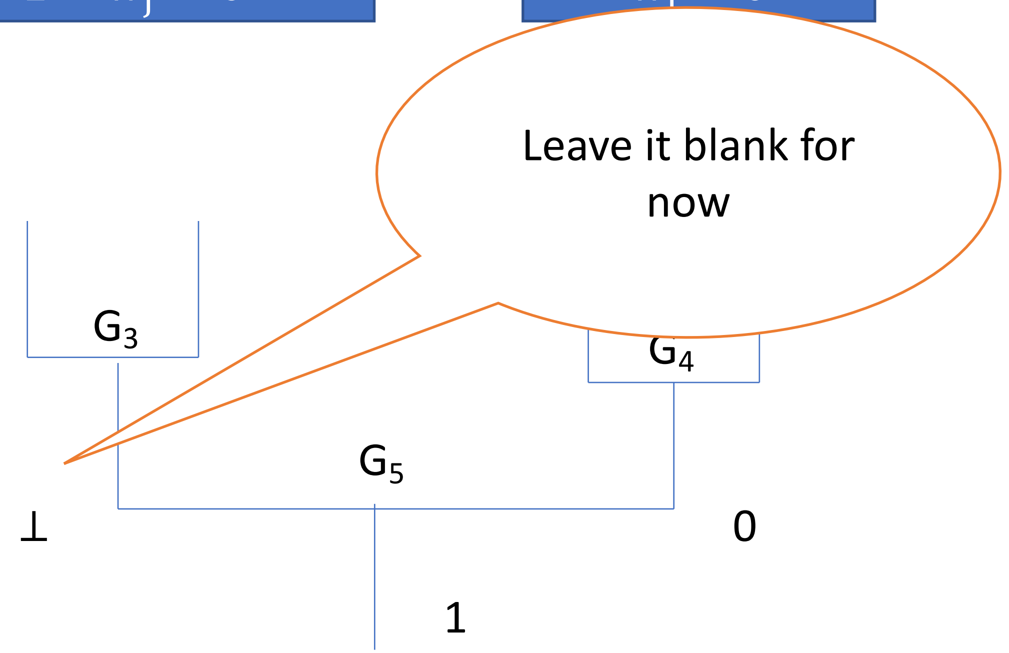


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_i = 0$

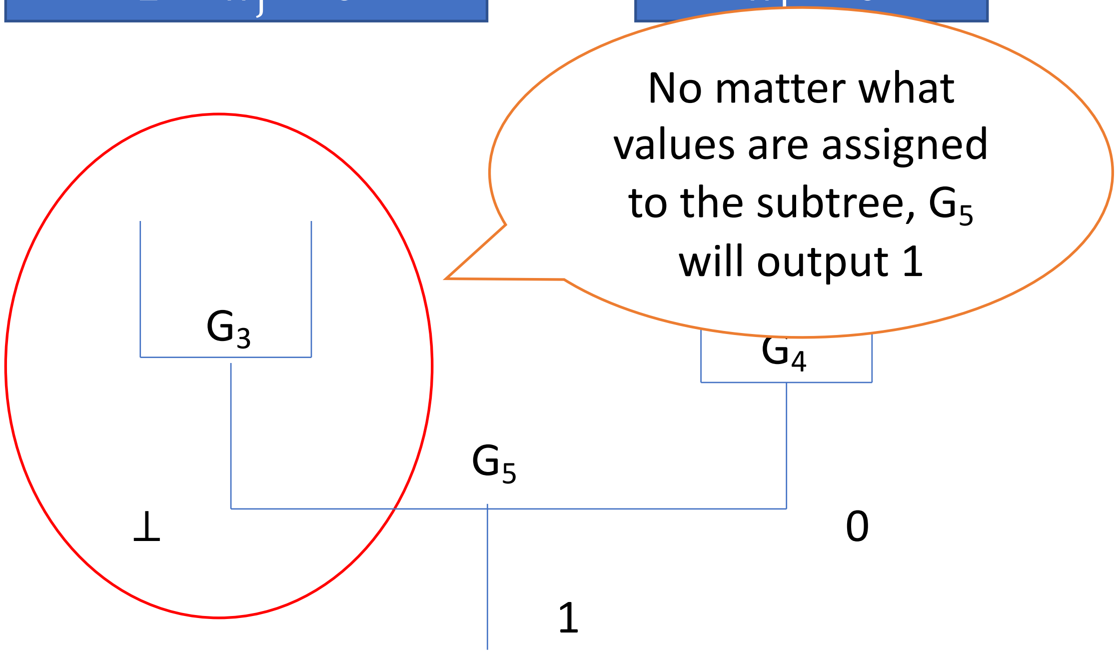


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_i = 0$

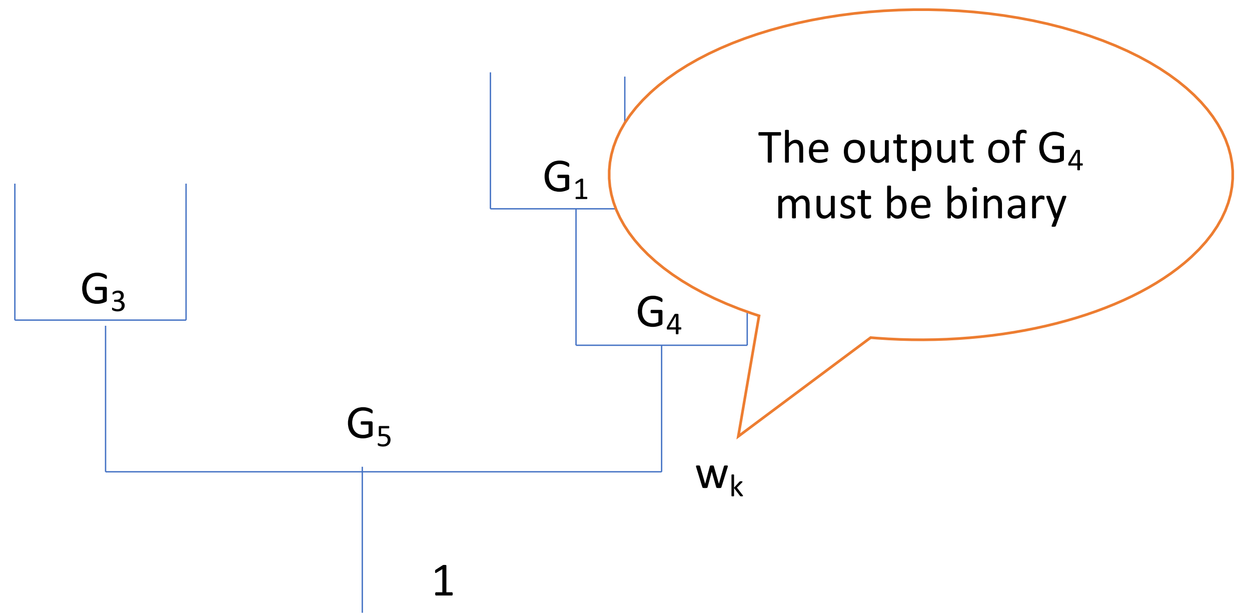


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$



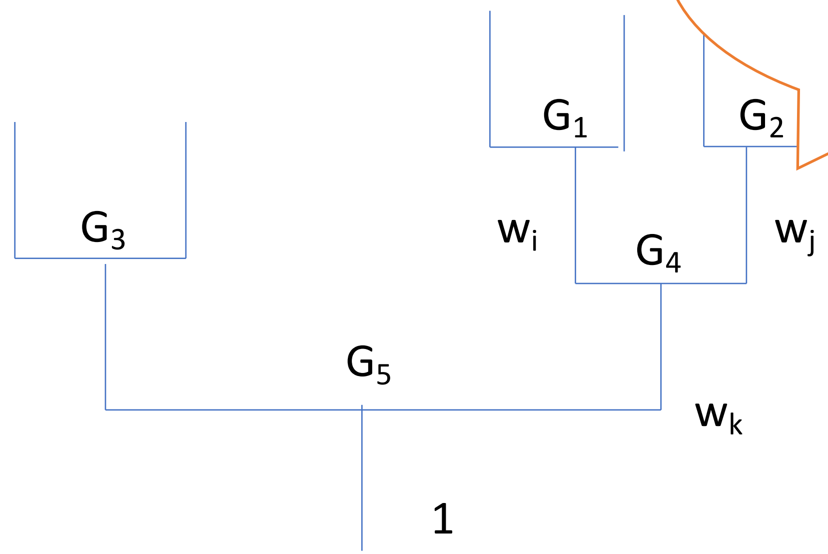
New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j =$

We assign values to its input wire(s) in a similar way

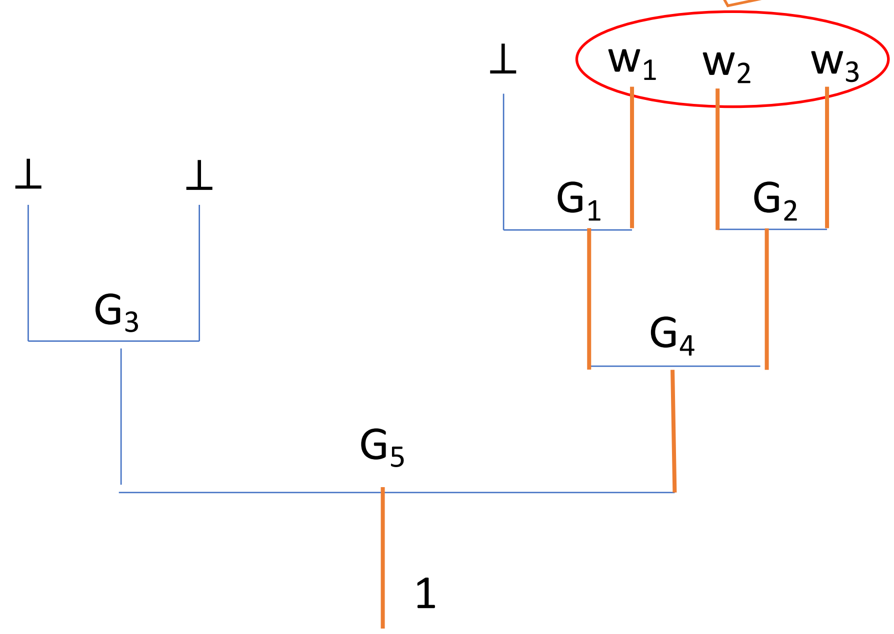


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

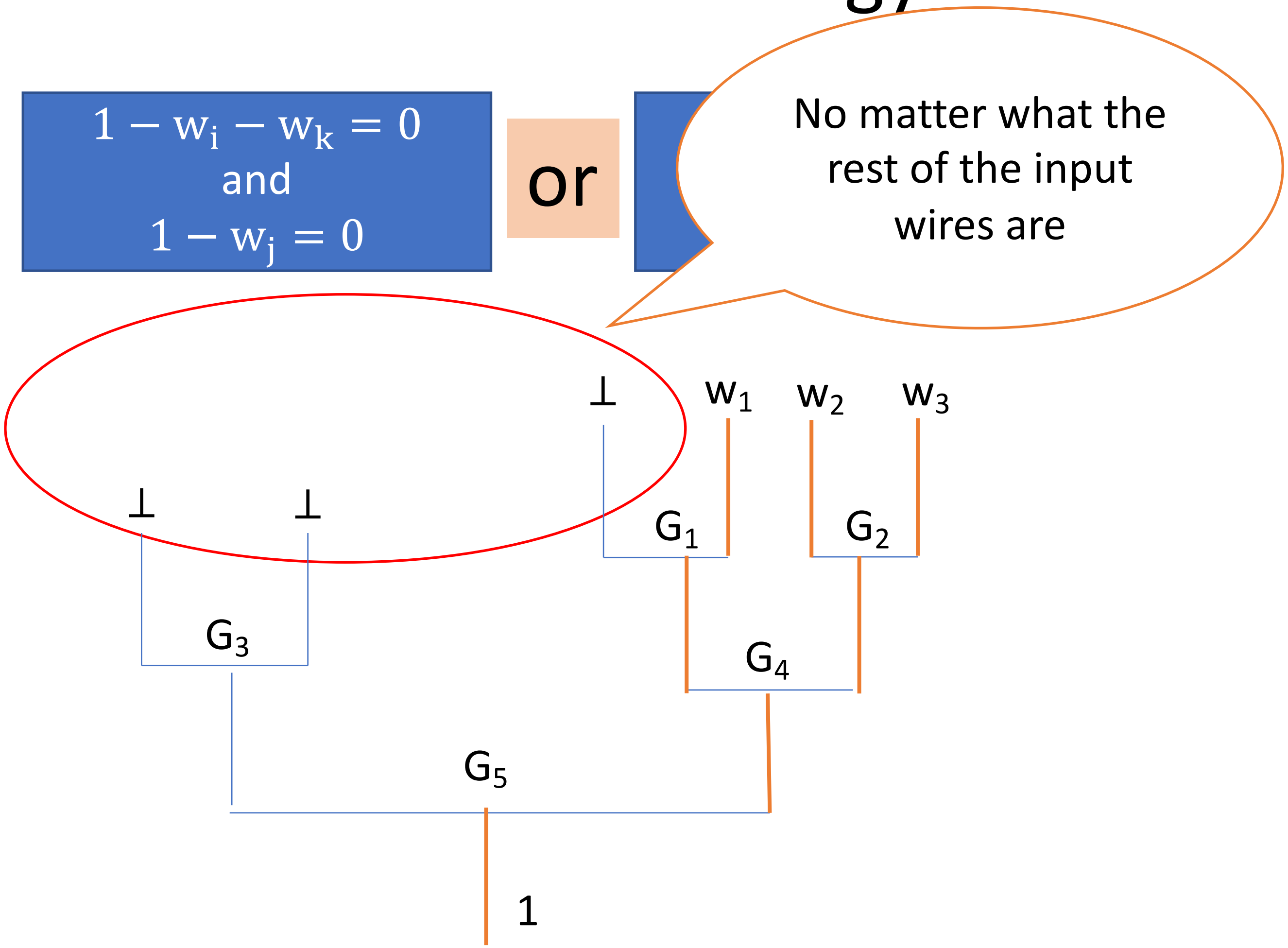
Recursively, we obtain part of the witness



New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

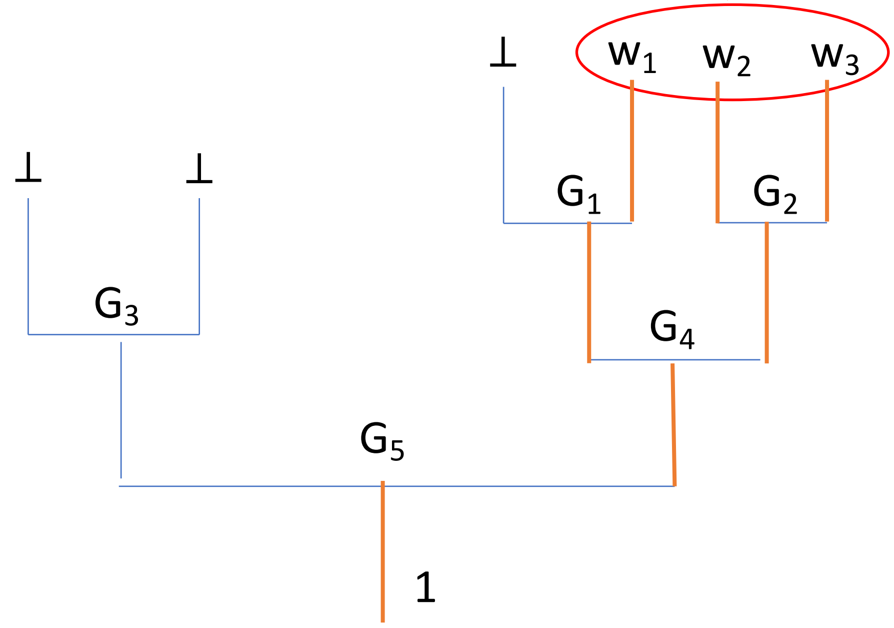


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

Assigned values will lead the circuit to output 1 anyway

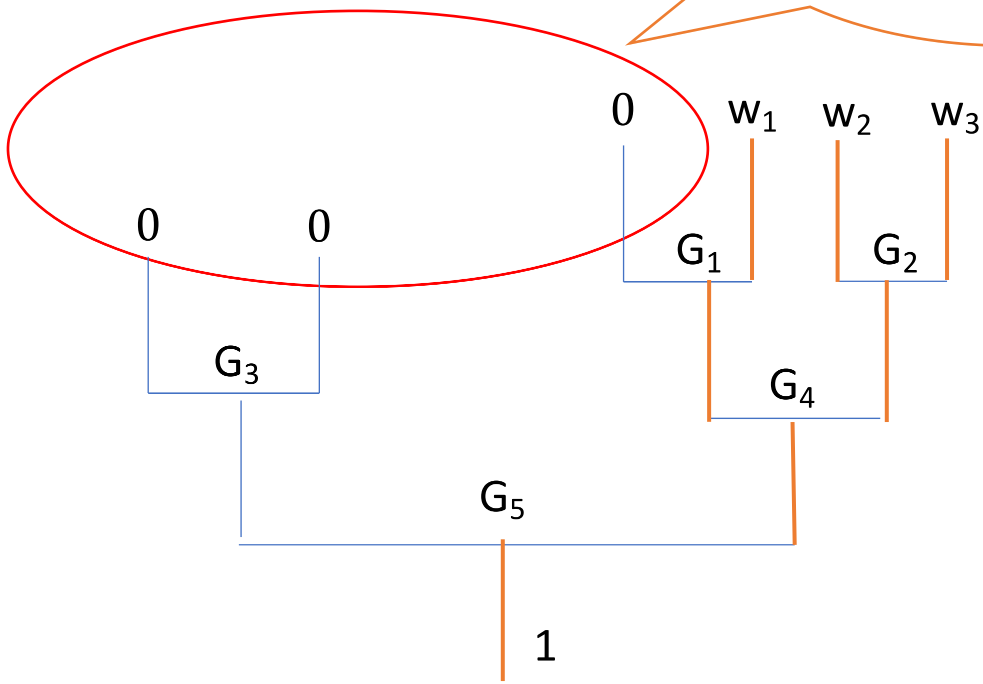


New Witness-Extraction Strategy

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

By setting the rest input wires as 0s, we obtain the witness

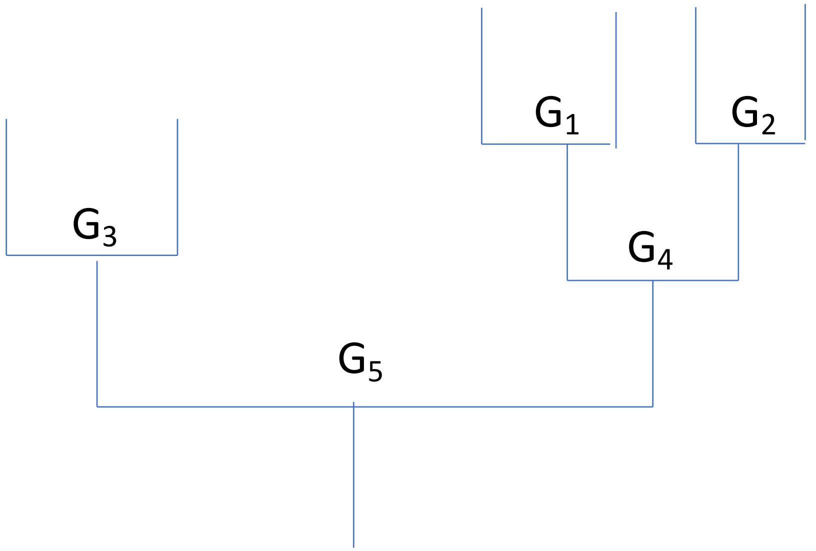


New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

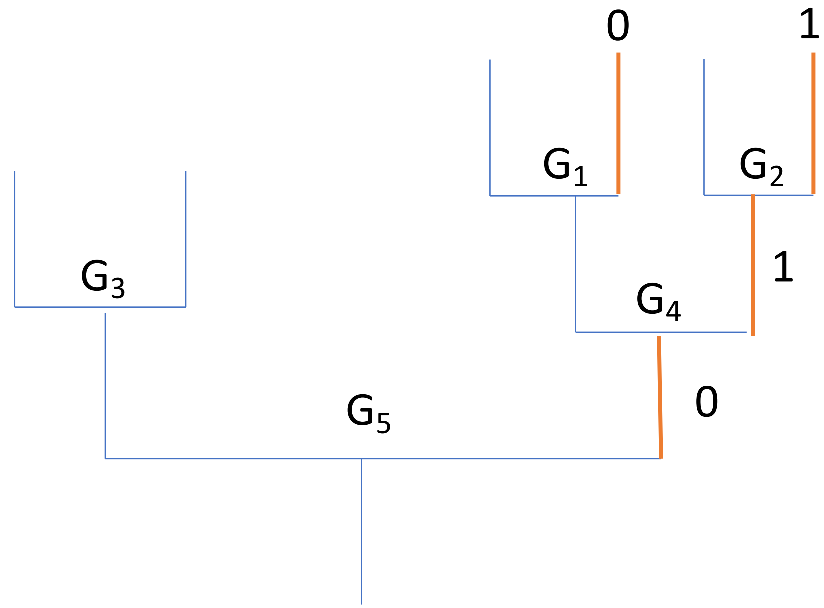


New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

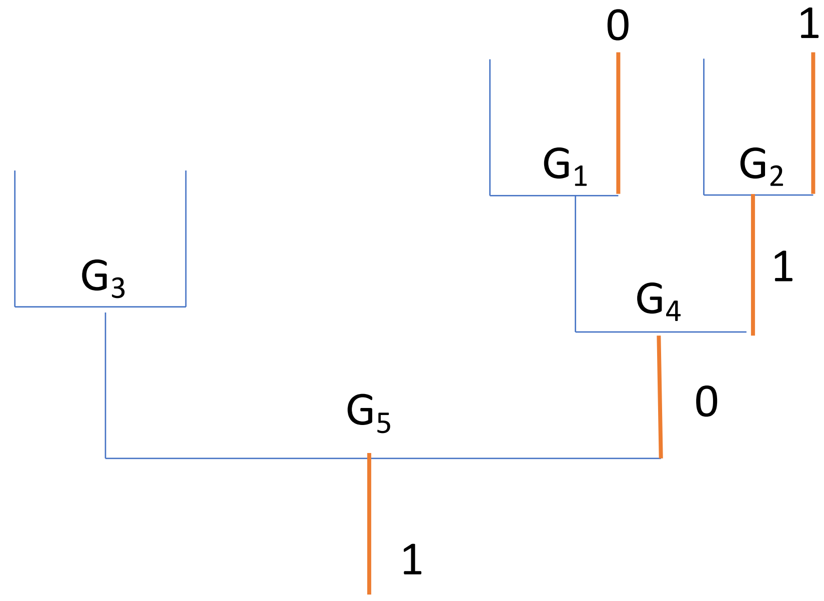


New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$

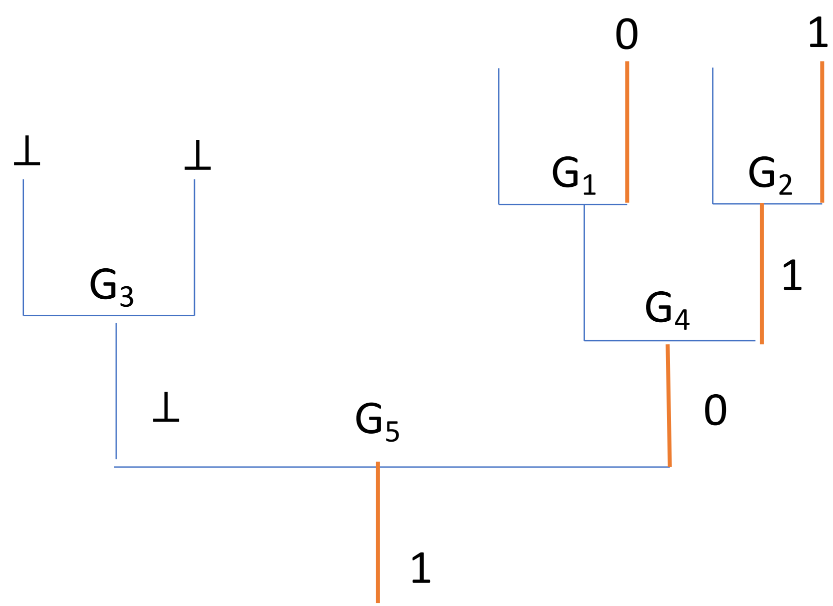


New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

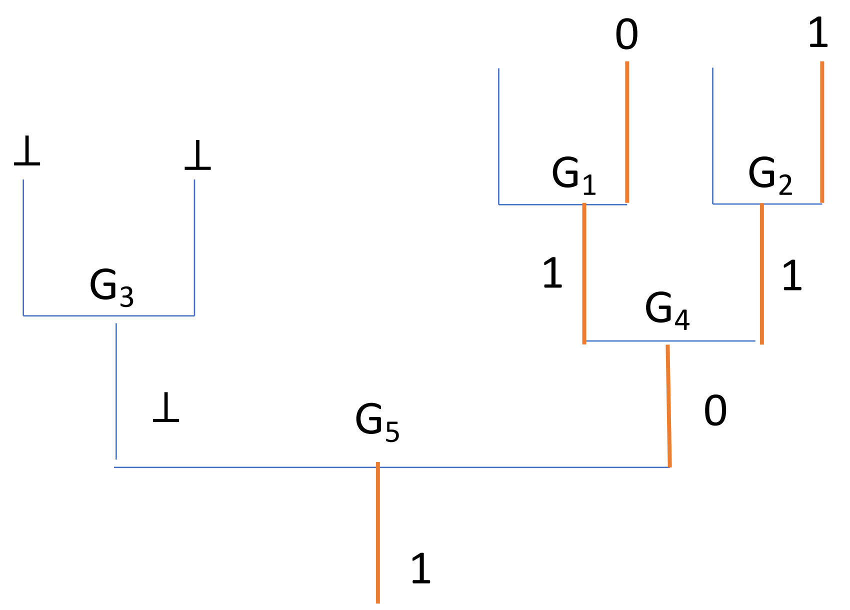
or

$1 - w_k = 0$
and
$w_j = 0$



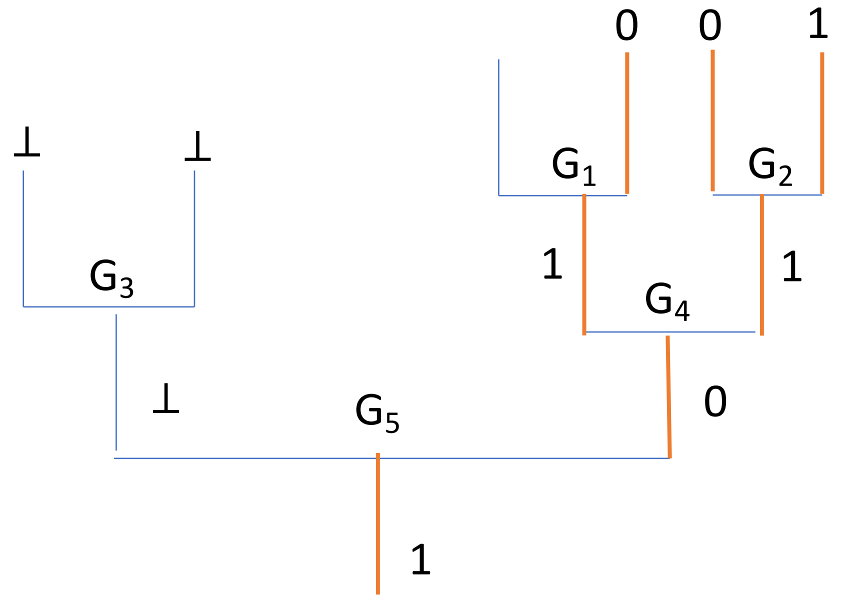
New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$ and $1 - w_j = 0$	or	$1 - w_k = 0$ and $w_j = 0$



New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$ and $1 - w_j = 0$	or	$1 - w_k = 0$ and $w_j = 0$

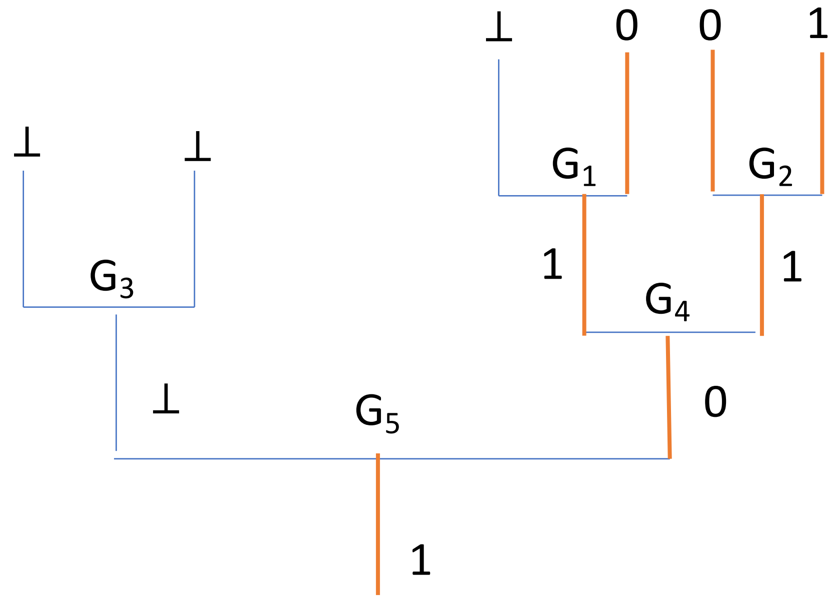


New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

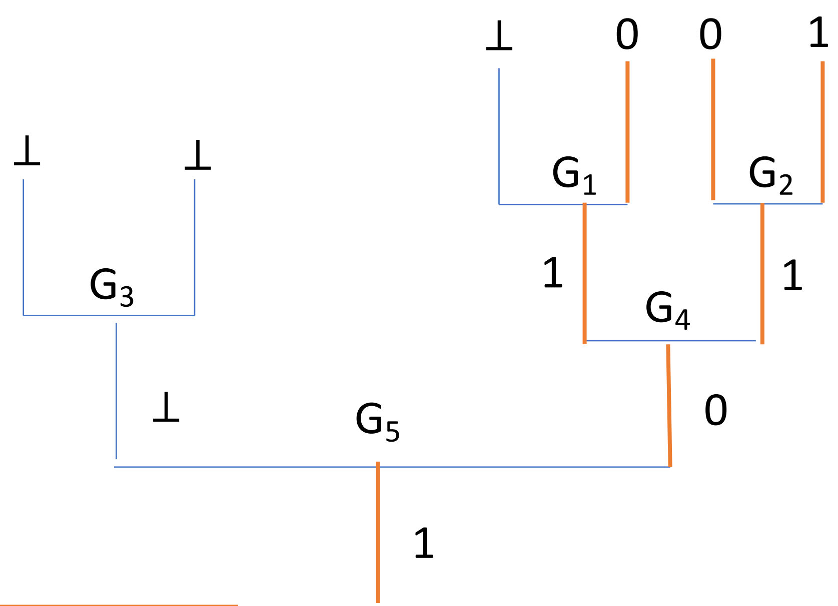
or

$1 - w_k = 0$
and
$w_j = 0$



New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$ and $1 - w_j = 0$	or	$1 - w_k = 0$ and $w_j = 0$



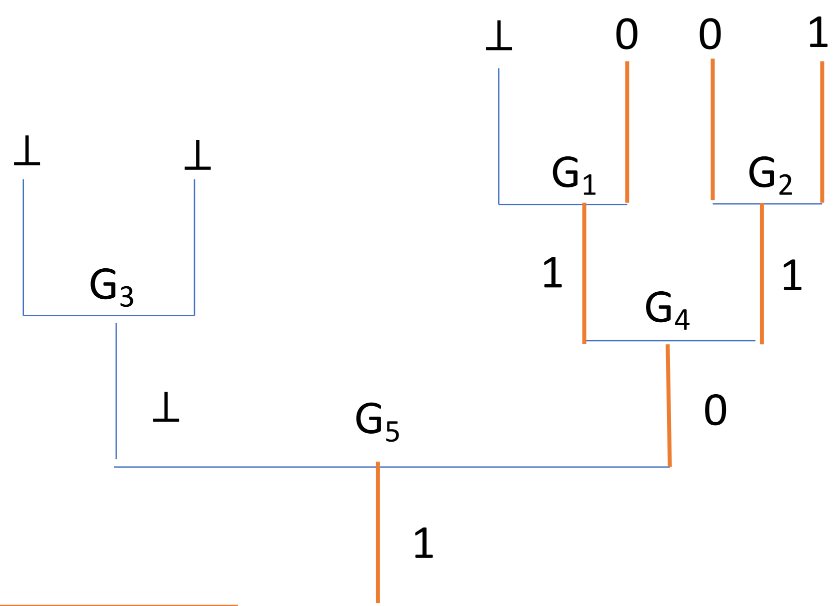
witness=(⊥, ⊥, ⊥,0,0,1)

New Witness-Extraction Strategy: Example

$1 - w_i - w_k = 0$
and
$1 - w_j = 0$

or

$1 - w_k = 0$
and
$w_j = 0$



witness=(⊥, ⊥, ⊥,0,0,1) → witness=(0,0,0,0,0,1)

Comparison: NIZK

Scheme	Sound.	ZK	CRS Size	Proof Size	Prov. Cost	Ver. Cost	Assump.
GOS12 [30] (sym. pair.)	comp. perf.	perf. comp.	$5\|\mathbb{G}\|$	$(9t+6s)\|\mathbb{G}\|$	$15t+12s$	$18(s+t)$	DLIN
GOS12* (asym. pair.)	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(6t+4s)\|\mathbb{G}_1\|+(6t+6s)\|\mathbb{G}_2\|$	$18t+16s$	$12(s+t)$	SXDH
Ours	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(2t+8s)\|\mathbb{G}_1\|+10s\|\mathbb{G}_2\|$	$2t+30s$	$24s$	SXDH

t: number of wires
s: number of gates
(t must larger than s)

Comparison: NIZK

Scheme	Sound.	ZK	CRS Size	Proof Size	Prov. Cost	Ver. Cost	Assump.
GOS12 [30] (sym. pair.)	comp. perf.	perf. comp.	$5\|\mathbb{G}\|$	$(9t+6s)\|\mathbb{G}\|$	$15t+12s$	$18(s+t)$	DLIN
GOS12* (asym. pair.)	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(6t+4s)\|\mathbb{G}_1\|+(6t+6s)\|\mathbb{G}_2\|$	$18t+16s$	$12(s+t)$	SXDH
Ours	comp. perf.	perf. comp.	$4\|\mathbb{G}_1\|+4\|\mathbb{G}_2\|$	$(2t+8s)\|\mathbb{G}_1\|+10s\|\mathbb{G}_2\|$	$2t+30s$	$24s$	SXDH

Our proof size and proving and verification cost are strictly smaller than GOS-NIZK

Comparison: Experimental Performance

When the ratio between number of gates and wires is 2, our proof size is about 1.62X smaller

Scheme	Proof Size (MB) (Ratio: 2.00)					Proof Size (MB) (Ratio: 1.50)					Proof Size (MB) (Ratio: 1.06)				
	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
GOS12 [30]	0.61	1.22	2.44	4.87	9.75	0.50	1.01	2.01	4.03	8.06	0.41	0.82	1.64	3.29	6.58
Ours	0.37	0.75	1.50	3.00	6.00	0.36	0.73	1.45	2.90	5.81	0.35	0.70	1.41	2.82	5.65

Comparison: Experimental Performance

Scheme	Ratio	Proving Cost (seconds)					Verification Cost (seconds)				
		2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
GOS12 [30]	2.00	1.38	2.69	5.39	10.81	21.72	12.55	25.80	50.57	101.11	201.95
Ours		0.87	1.82	3.51	6.99	14.37	8.68	17.38	37.23	70.04	138.70
GOS12 [30]	1.50	1.17	2.23	[illegible]	9.27	17.87	10.61	21.15	[illegible]	84.91	168.13
Ours		0.85	[illegible]	[illegible]	6.74	13.75	8.61	17.27	[illegible]	68.60	141.79
GOS12 [30]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	14.65	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Ours		[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Our prover is about 1.52X faster

Our verifier is about 1.44X faster

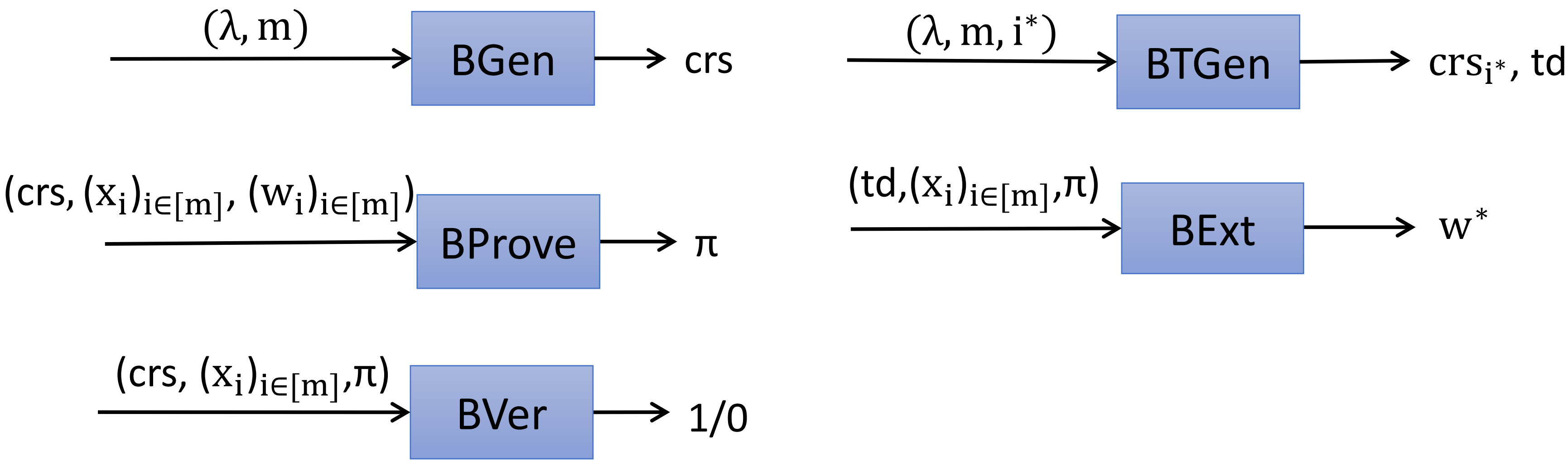
Proof systems

Non-interactive zero-knowledge proof (NIZK)

Non-interactive batch argument (BARG)

Definition of BARG for NP

$$L_m^{\mathrm{BatchCSAT}} = \{C \mid \forall i \in [m]: \exists w_i: C(w_i) = 1\}$$



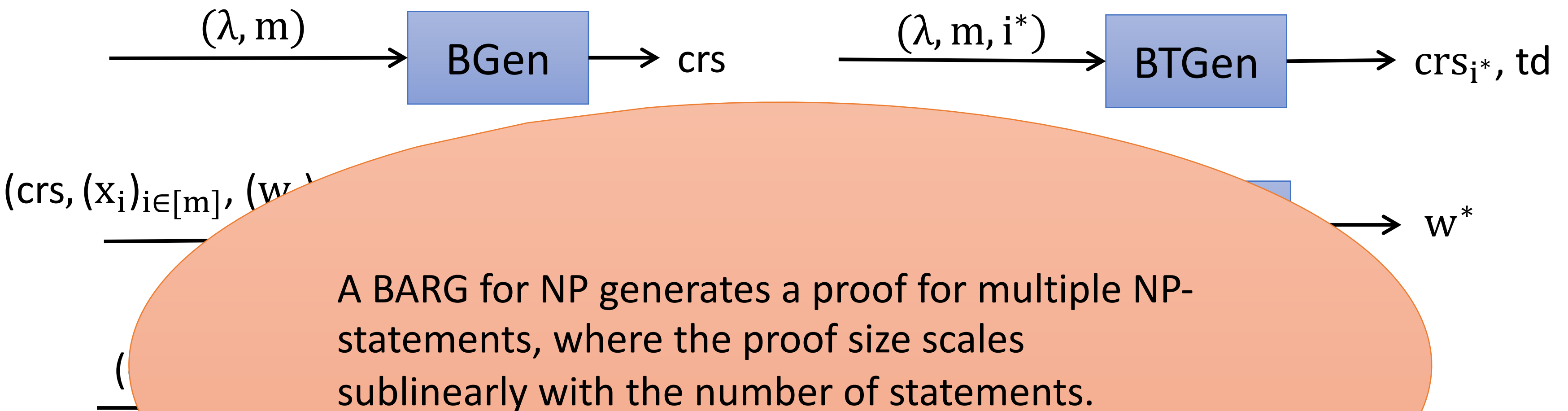
Completeness: honest proofs must pass the verification.

Succinctness: the proof size, crs size, and verification running time is succinct. Here, our proof size is independent of m.

Somewhere argument of knowledge: crs and crs_{i^*} are indistinguishable, and when in the trapdoor mode, BExt is able to extract a valid witness for x_{i^*} for any valid statement/proof pair $((x_i)_{i\in[m]},\pi)$.

Definition of BARG for NP

$$L_m^{BatchCSAT} = \{C \mid \forall i \in [m]: \exists w_i: C(w_i) = 1\}$$



A BARG for NP generates a proof for multiple NP-statements, where the proof size scales sublinearly with the number of statements.

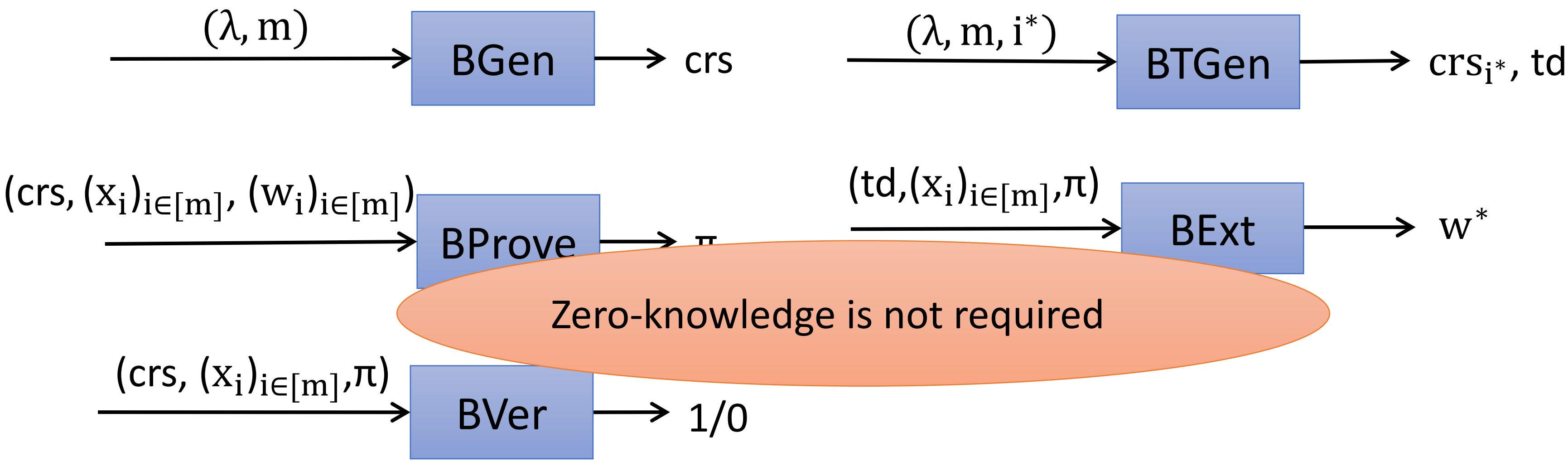
Complete

Succinctness: the proof size, crs size, and verification running time is succinct. Here, our proof size is independent of m.

Somewhere argument of knowledge: crs and crs_{i^*} are indistinguishable, and when in the trapdoor mode, BExt is able to extract a valid witness for x_{i^*} for any valid statement/proof pair $((x_i)_{i\in[m]},\pi)$.

Definition of BARG for NP

$$L_m^{\mathrm{BatchCSAT}} = \{C \mid \forall i \in [m]: \exists w_i: C(w_i) = 1\}$$



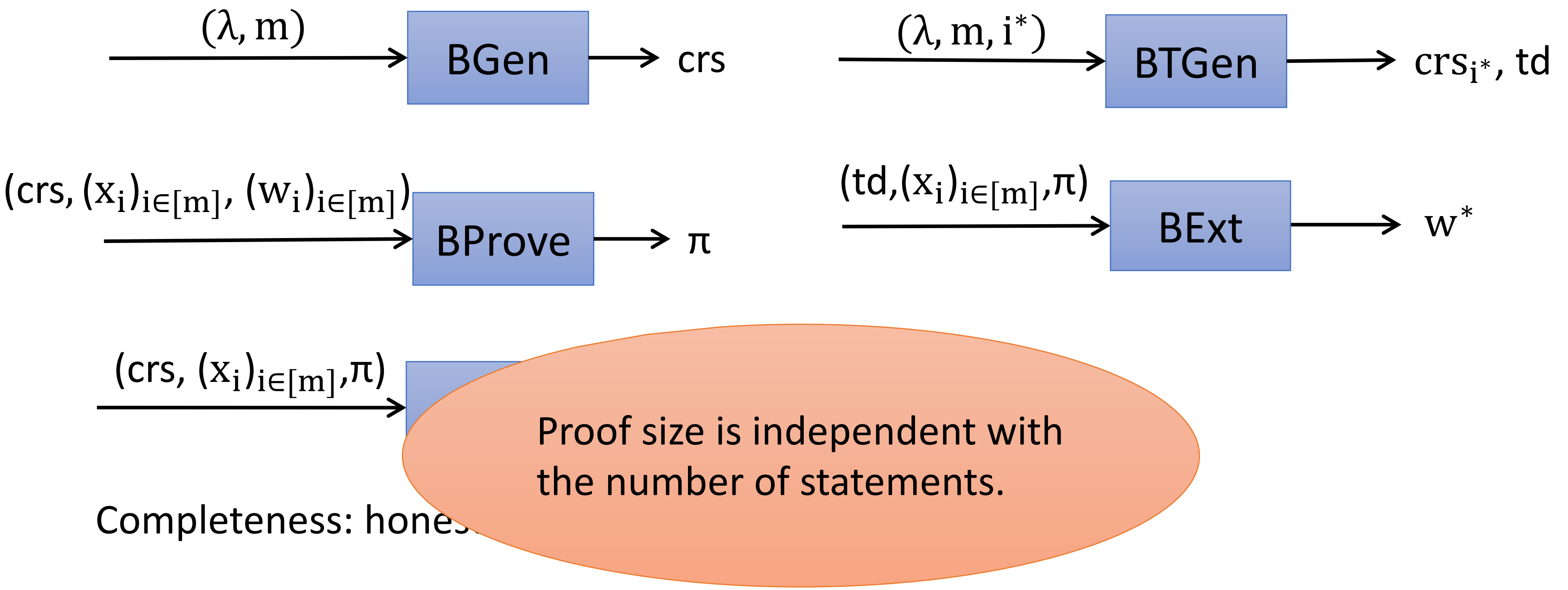
Completeness: honest proofs must pass the verification.

Succinctness: the proof size, crs size, and verification running time is succinct. Here, our proof size is independent of m.

Somewhere argument of knowledge: crs and crs_{i^*} are indistinguishable, and when in the trapdoor mode, BExt is able to extract a valid witness for x_{i^*} for any valid statement/proof pair $((x_i)_{i\in[m]},\pi)$.

Definition of BARG for NP

$$L_m^{\mathrm{BatchCSAT}} = \{C \mid \forall i \in [m]: \exists w_i: C(w_i) = 1\}$$



Completeness: hones

Succinctness: the proof size, crs size, and verification running time is succinct. Here, our proof size is independent of m.

Somewhere argument of knowledge: crs and crs_{i^*} are indistinguishable, and when in the trapdoor mode, BExt is able to extract a valid witness for x_{i^*} for any valid statement/proof pair $((x_i)_{i\in[m]}, \pi)$.

Existing BARG for NP

Assumptions:

- Both quadratic residuosity assumption and the subexponentially hard Diffie-Hellman assumption, learning with errors assumption[CJJ21a,CJJ21b]
- MDDH assumption, subgroup decision [WW22]
- Non-standard assumptions[KPY19]
- Non-falsifiable assumptions[Gro10, BCcm12, DFH12, Lip13, PHGR13, GGPR13, BCI+13, BCPR14, BISW17, BCC+17]
- Idealized models[Mic95, Gro16, BBHR18, COS20, CHM 20, Set20]

Existing BARG for NP

Assumptions:

- Both quadratic residuosity assumpti subexponentially hard Diffie-Hellman as with errors assumption[CJJ21a,CJJ21b]
- MDDH assumption, subgroup decision [WW22]
- Non-standard assumptions[KPY19]
- Non-falsifiable assumptions[Gro10, BCcm12, DFH12, Lip13, PHGR13, GGPR13, BCI+13, BCPR14, BISW17, BCC+17]
- Idealized models[Mic95, Gro16, BBHR18, COS20, CHM 20, Set20]

State-of-the-art in pairings

Our Results

Pairing-based BARGs for NP with shorter proofs and less proving and verification cost than WW-BARG.

Assumption: MDDH assumption or subgroup decision assumption

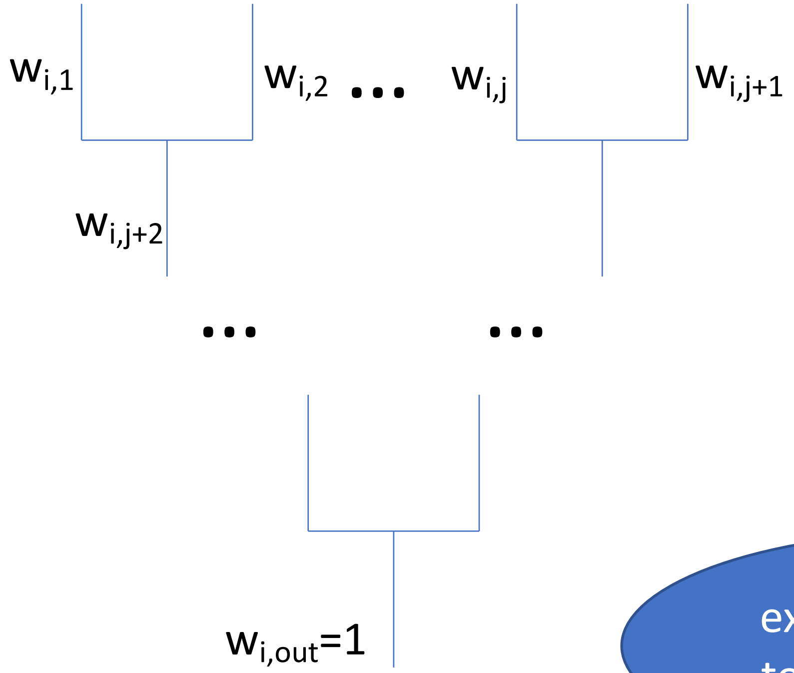
Our Results

Pairing-based BARGs for NP with shorter proofs and less proving and verification cost than WW-BARG.

No trade-off

BARG for NP [WW22]

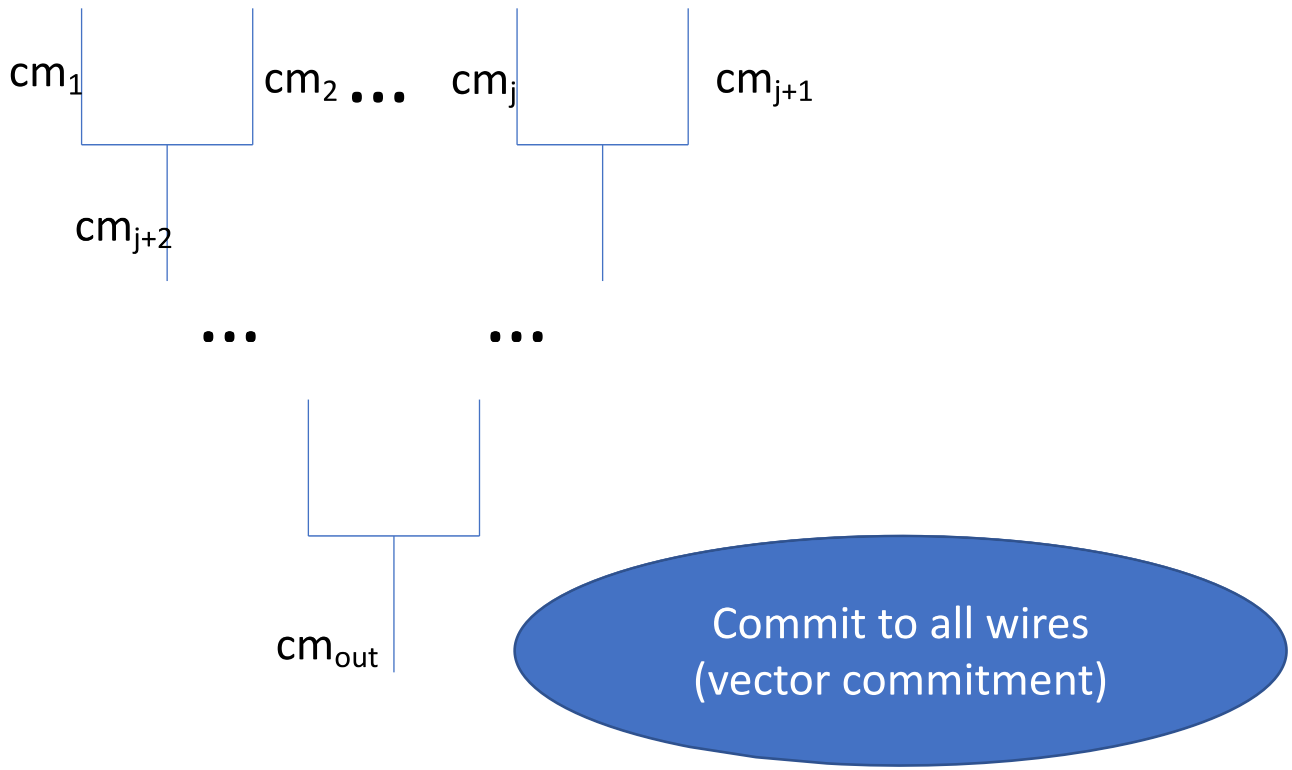
Prover:



The prover first extends the witness to contain bits of all wires

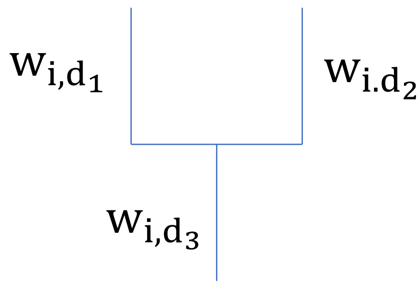
BARG for NP [WW22]

Prover:



BARG for NP [WW22]

Prover:



NAND gate

The prover generates succinct proofs of wire validity and gate consistency.

For all $i \in [m]$,
$1 - w_{i,d_1} w_{i.d_2} = w_{i,d_3}$

gate consistency

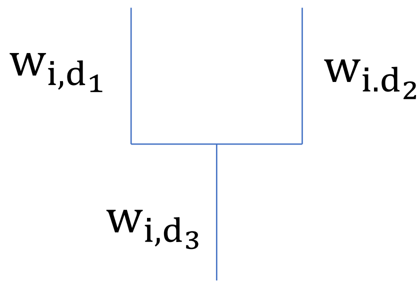
and

For all $i \in [m], j \in [t]$,
$w_{i,j}$ = 0 or 1

wire validity

BARG for NP [WW22]

Prover:



NAND gate

If we can prove gate consistency for the relation used by our NIZK, we can reduce the cost

The prover generates succinct proofs of wire val… …ncy.

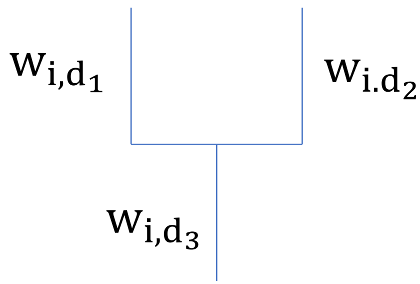
For all $i \in [m]$,
$1 - w_{i,d_1} w_{i.d_2} = w_{i,d_3}$

and

For all $i \in [m], j \in [t]$,
$w_{i,j}$ = 0 or 1

BARG for NP [WW22]

Prover:



NAND gate

We do not have an explicit "batch OR-proof".

The prover generates succinct proofs of wire validity and gate consistency.

For all $i \in [m]$,
$1 - w_{i,d_1} w_{i.d_2} = w_{i,d_3}$

and

For all $i \in [m], j \in [t]$,
$w_{i,j}$ = 0 or 1

Solution

For all $i \in [m]$,
$(1 - w_{i,d_1} - w_{i,d_3})w_{i,d_2} = 0$

and

For all $i \in [m]$,
$(1 - w_{i,d_3})(1 - w_{i,d_2}) = 0$

Prove non-linear relations for each NAND gate

Solution

For all $i \in [m]$,
$(1 - w_{i,d_1} - w_{i,d_3})w_{i,d_2} = 0$

and

More "relaxed" version of OR-relations for witnesses

For all $i \in [m]$,
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

For all $i \in [m]$,
$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

For all $i \in [m]$,
$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$

Solution

For all $i \in [m]$,
$(1 - w_{i,d_1} - w_{i,d_3})w_{i,d_2} = 0$

and

For all $i \in [m]$,
$w_{i,d_2}) = 0$

Generalized witness-extraction strategy

For all $i \in [m]$,
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

and
$w_{i,d_2} = 0$

or

For all $i \in [m]$,
$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$

New Witness-Extraction Strategy

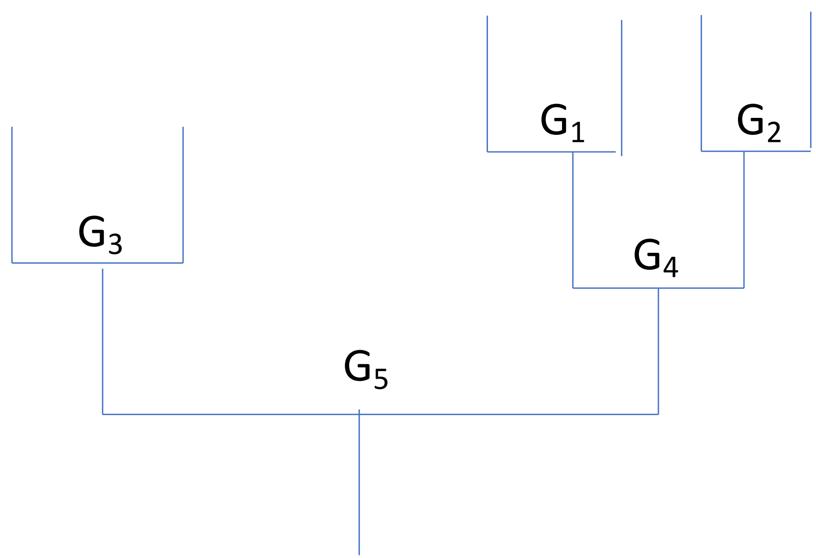
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



New Witness-Extraction Strategy: Examples

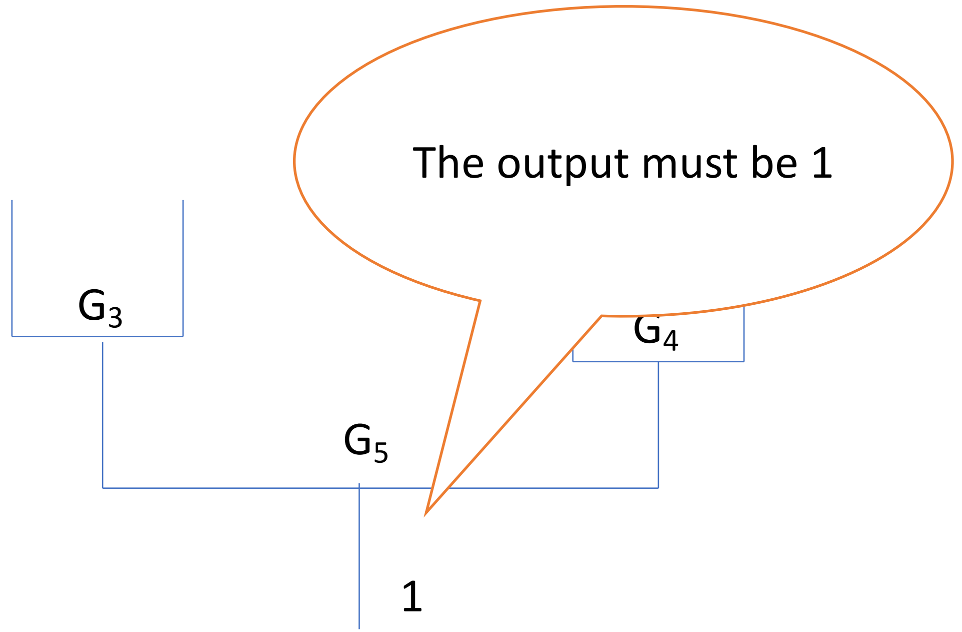
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



New Witness-Extraction Strategy: Examples

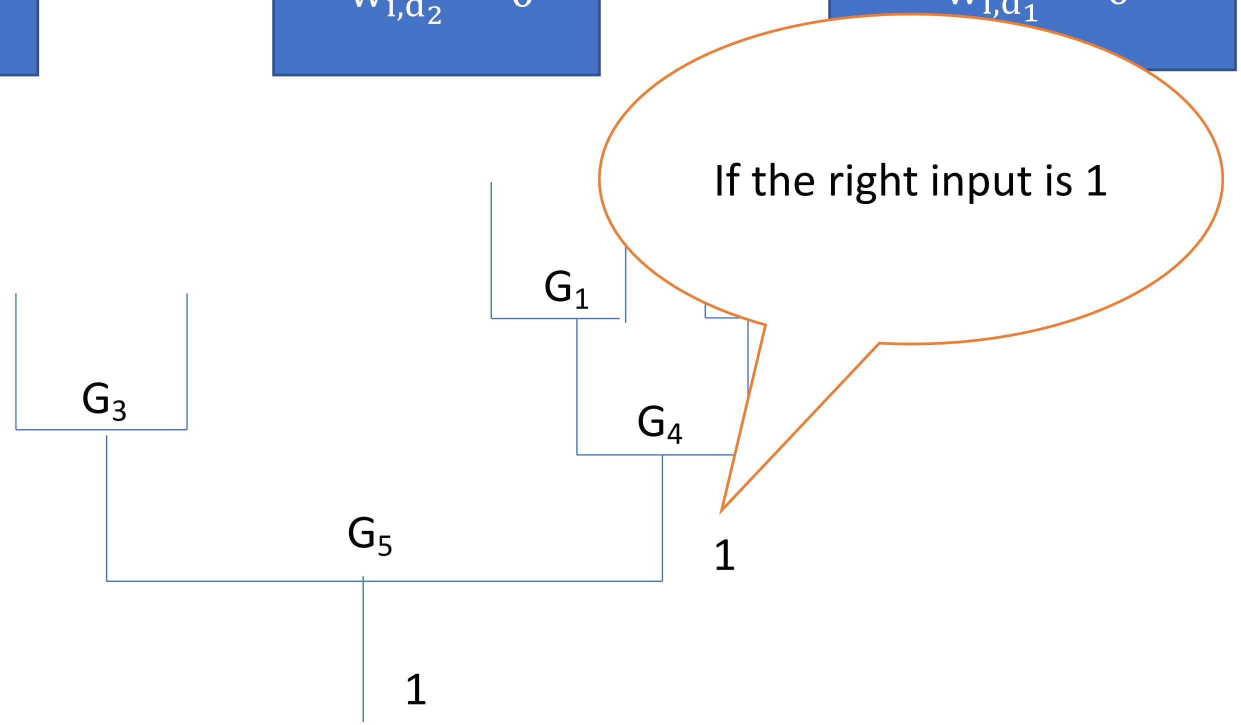
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



New Witness-Extraction Strategy: Examples

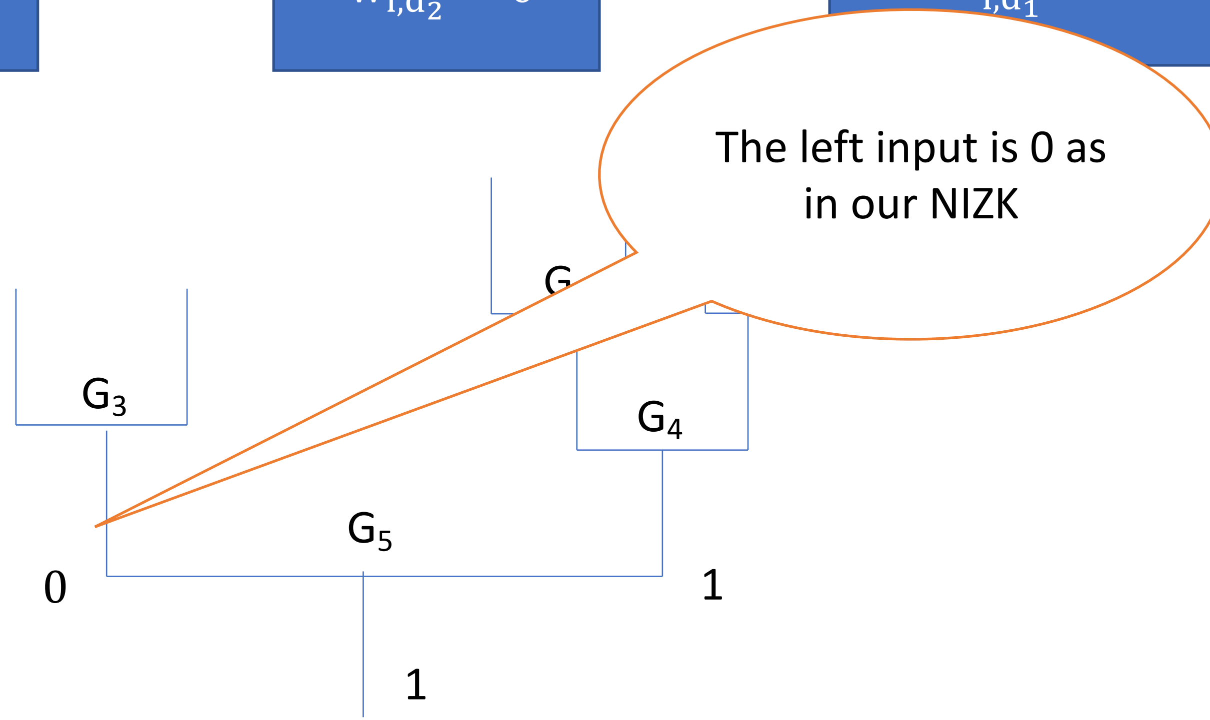
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



New Witness-Extraction Strategy: Examples

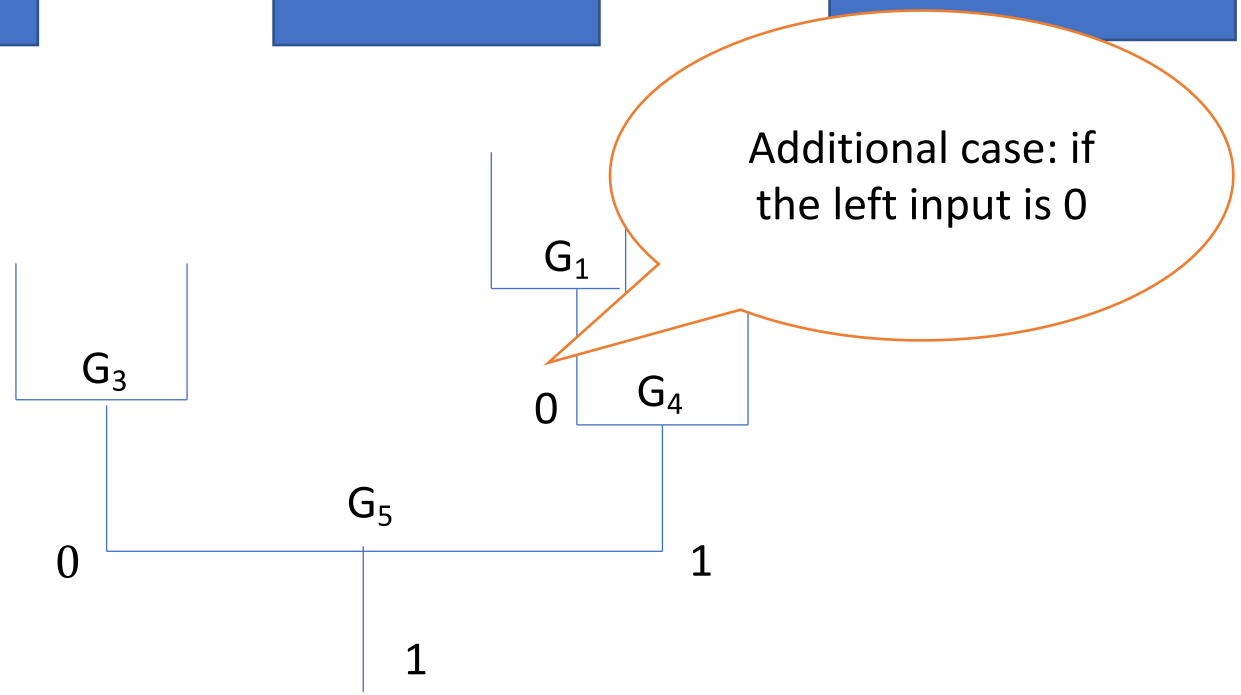
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



New Witness-Extraction Strategy: Examples

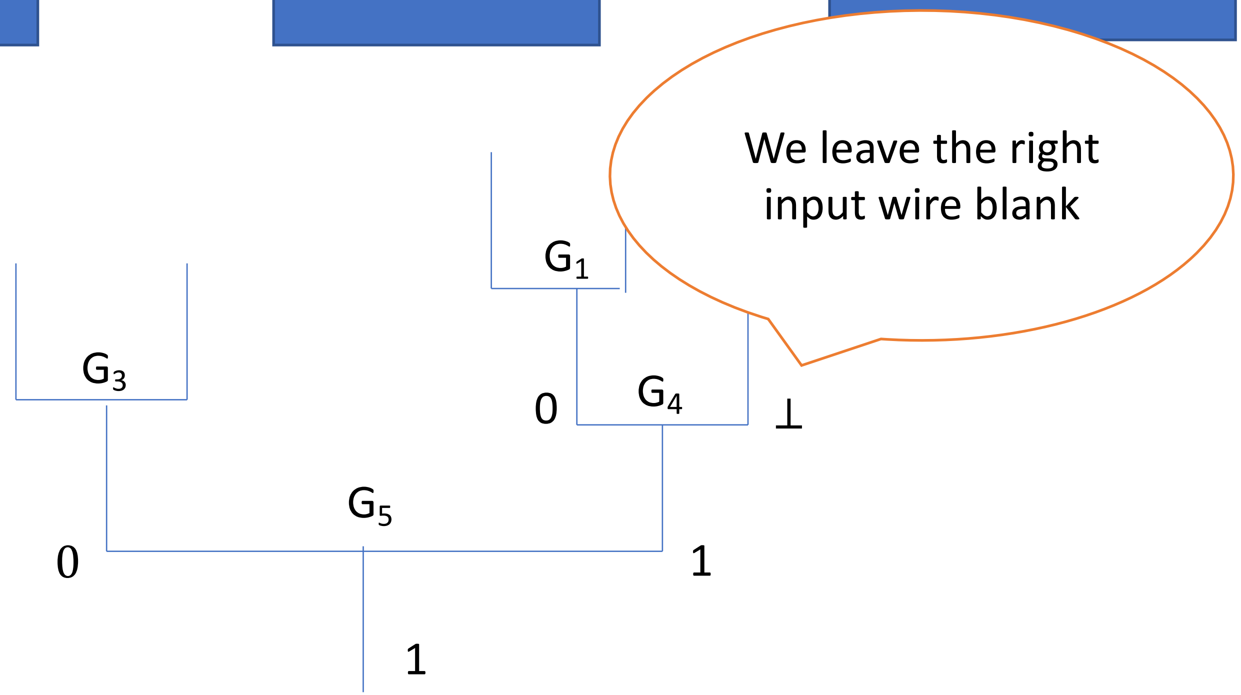
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_1} = 0$



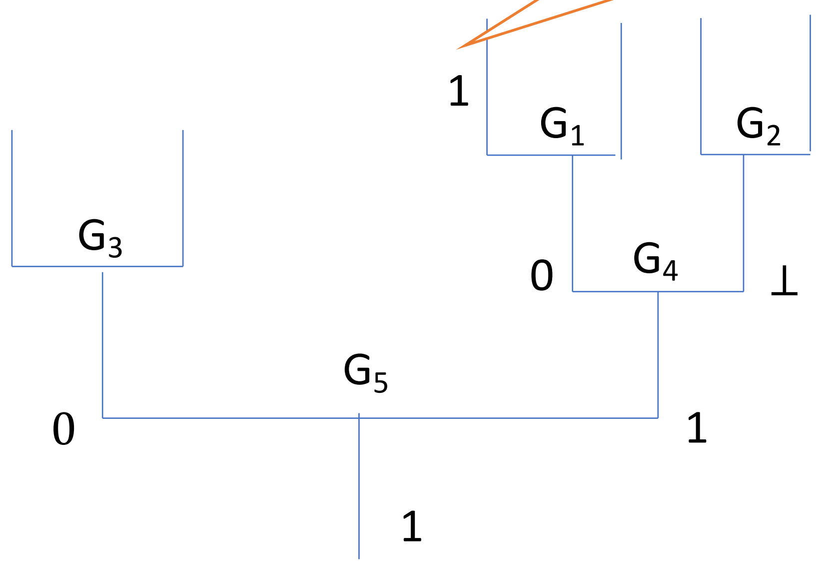
New Witness-Extraction Strategy: Examples

$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

Continue to extract the values for G_1



New Witness-Extraction Strategy: Examples

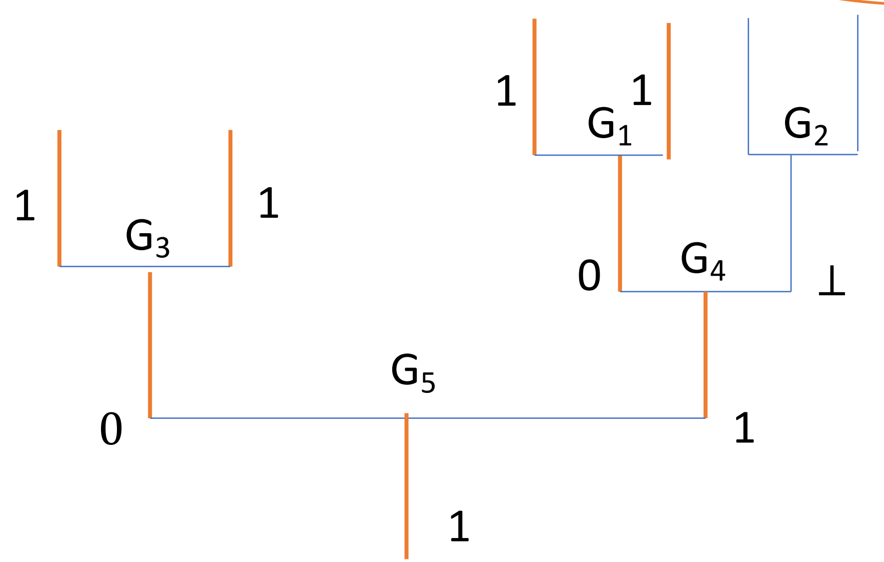
$1 - w_{i,d_1} - w_{i,d_3} = 0$
and
$1 - w_{i,d_2} = 0$

or

$1 - w_{i,d_3} = 0$
and
$w_{i,d_2} = 0$

$= 0$

Recursively, we obtain part of the witness leading the circuit to output 1



Comparison: BARG

Scheme	CRS Size	Proof Size	Prov. Cost	Ver. Cost	Assump.
WW22 [49] (asym. pair.)	$(4+2m^2)\|\mathbb{G}_1\|+ (4+2m^2)\|\mathbb{G}_2\|$	$(4t+4s)\|\mathbb{G}_1\| + (4t+4s)\|\mathbb{G}_2\|$	$4m^2t+4m(m-1)s$	$24t+32s$	SXDH
WW22* [49] (sym. pair.)	$(1+m^2)\|\mathbb{G}\|$	$(2t+s)\|\mathbb{G}\|$	$m^2t+\frac{m(m-1)}{2}s$	$2t+3s$	Subgroup decision
Ours (asym. pair.)	$(4+2m^2)\|\mathbb{G}_1\|+ (4+2m^2)\|\mathbb{G}_2\|$	$(2t+6s)\|\mathbb{G}_1\| + (2t+6s)\|\mathbb{G}_2\|$	$4mt+6m(m-1)s$	$40s$	SXDH
Ours (sym. pair.)	$(1+m^2)\|\mathbb{G}\|$	$(t+2s)\|\mathbb{G}\|$	$mt+m(m-1)s$	$4s$	Subgroup decision

Our proof size and proving and verification cost are strictly smaller in both prime- and composite-order groups.

Comparison: Experimental Performance

Scheme	Proof Size (MB) (Ratio: 2.00)					Proof Size (MB) (Ratio: 1.50)					Proof Size (MB) (Ratio: 1.06)				
	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
WW22 [49] (100 stats.)	0.42	0.84	1.69	3.37	6.75	0.35	0.70	1.41	2.81	5.62	0.29	0.58	1.16	2.32	4.64
Ours (100 stats.)	0.35	0.70	1.41	2.81	5.62	0.32	0.63	1.26	2.53	5.06	0.28	0.57	1.14	2.28	4.57
WW22 [49] (50 stats.)	0.42	0.84	1.69	3.37	6.75	0.35	0.70	1.41	2.81	5.62	0.29	0.58	1.16	2.32	4.64
Ours (50 stats.)	0.35	0.70	1.41	2.81	5.62	0.32	0.63	1.26	2.53	5.06	0.28	0.57	1.14	2.28	4.57

When the ratio between number of gates and wires is 2, our proof size is 1.20x smaller

Comparison: Experimental Performance

Scheme	Ratio	Proving Cost (seconds)					Verification Cost (seconds)				
		2^8	2^9	2^{10}	2^{11}	2^{12}	2^8	2^9	2^{10}	2^{11}	2^{12}
WW22 [49] (100 stats.)	2.00	2.50	4.64	9.93	18.36	37.44	15.69	30.23	65.45	123.66	255.95
Ours (100 stats.)		1.07	2.02	4.10	8.00	16.91	5.90	11.61	23.38	46.41	94.46
WW22 [49] (50 stats.)	2.00	0.61	1.22	2.4[illegible]	4.71	9.74	16.43	31.16	6[illegible].21	118.37	253.20
Ours (50 stats.)		0.29	0.5[illegible]	[illegible].20	2.05	4.67	5.68	11.44	[illegible]40	46.56	95.28

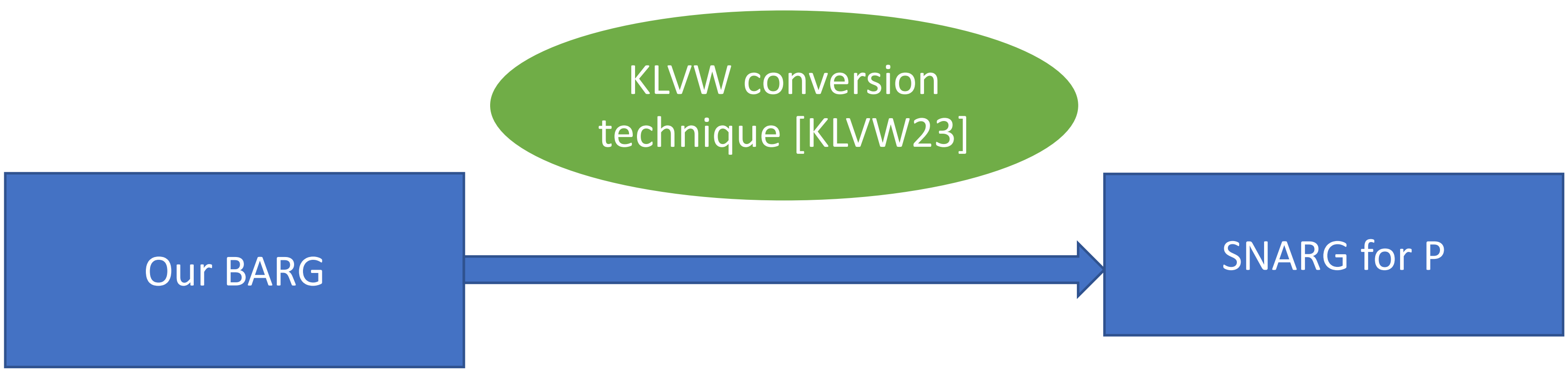
When proving 100 statements, our prover is about 2.27x faster

Our verifier is about 2.70x faster

Extensions

❖ Conversion to non-interactive zaps (NIWI in the plain model)

❖ Conversion to SNARG for P



Conclusion

A simple and efficient framework of proof systems for NP which improves the efficiency of GOS-NIZK and WW-BARG without any trade-off.