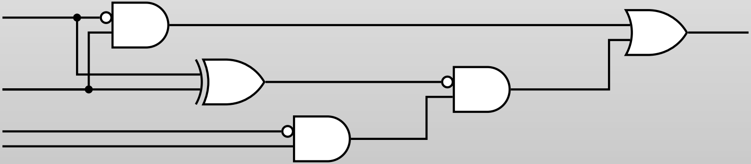
Improvements for **Gate-Hiding** Garbled Circuits

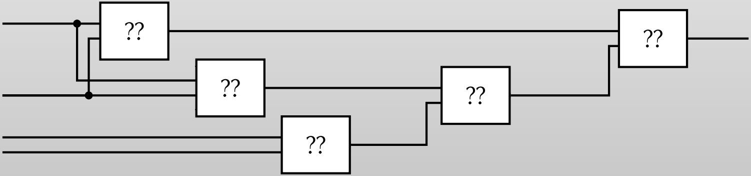
Mike Rosulek
Oregon State University OSU



ia.cr/2017/976

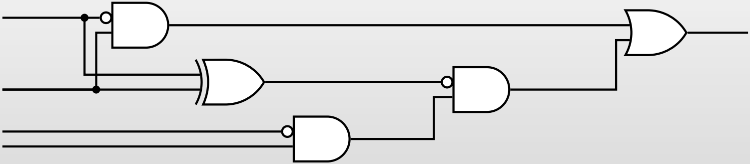
Improvements for
Gate-Hiding Garbled Circuits

Mike Rosulek
Oregon State University OSU

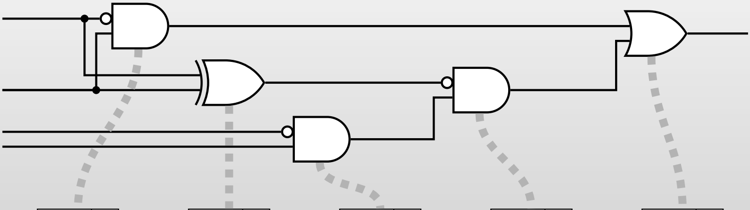


ia.cr/2017/976

Garbled circuit framework [Yao86]



Garbled circuit framework [Yao86]



0	0	0
0	1	1
1	0	0
1	1	0

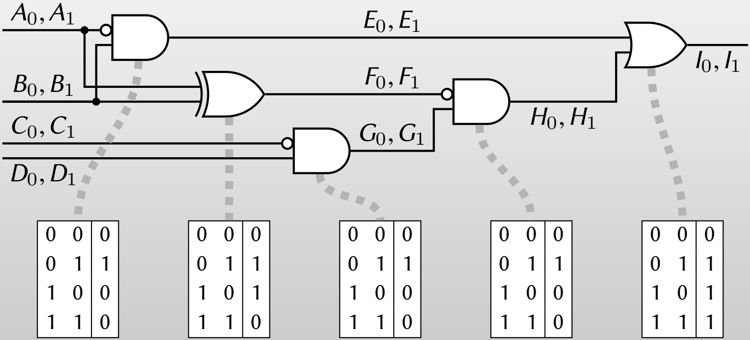
0	0	0
0	1	1
1	0	1
1	1	0

0	0	0
0	1	1
1	0	0
1	1	0

0	0	0
0	1	1
1	0	0
1	1	0

0	0	0
0	1	1
1	0	1
1	1	1

Garbled circuit framework [Yao86]



0	0	0
0	1	1
1	0	0
1	1	0

0	0	0
0	1	1
1	0	1
1	1	0

0	0	0
0	1	1
1	0	0
1	1	0

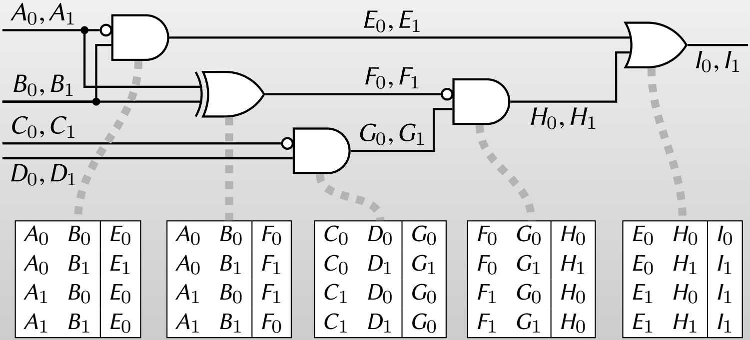
0	0	0
0	1	1
1	0	0
1	1	0

0	0	0
0	1	1
1	0	1
1	1	1

Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire

Garbled circuit framework [Yao86]



A_0	B_0	E_0
A_0	B_1	E_1
A_1	B_0	E_0
A_1	B_1	E_0

A_0	B_0	F_0
A_0	B_1	F_1
A_1	B_0	F_1
A_1	B_1	F_0

C_0	D_0	G_0
C_0	D_1	G_1
C_1	D_0	G_0
C_1	D_1	G_0

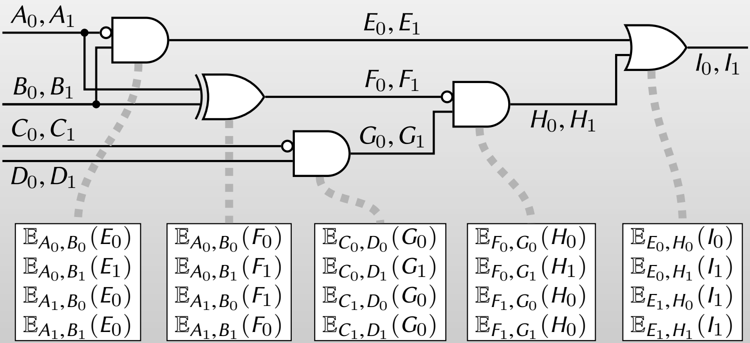
F_0	G_0	H_0
F_0	G_1	H_1
F_1	G_0	H_0
F_1	G_1	H_0

E_0	H_0	I_0
E_0	H_1	I_1
E_1	H_0	I_1
E_1	H_1	I_1

Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire

Garbled circuit framework [Yao86]



$\mathbb{E}_{A_0,B_0}(E_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$
$\mathbb{E}_{A_1,B_1}(E_0)$

$\mathbb{E}_{A_0,B_0}(F_0)$
$\mathbb{E}_{A_0,B_1}(F_1)$
$\mathbb{E}_{A_1,B_0}(F_1)$
$\mathbb{E}_{A_1,B_1}(F_0)$

$\mathbb{E}_{C_0,D_0}(G_0)$
$\mathbb{E}_{C_0,D_1}(G_1)$
$\mathbb{E}_{C_1,D_0}(G_0)$
$\mathbb{E}_{C_1,D_1}(G_0)$

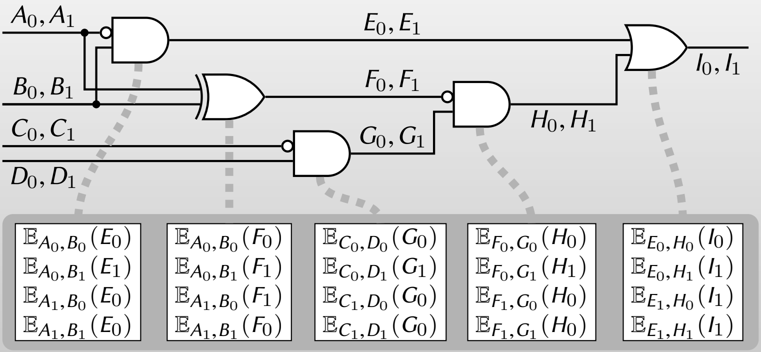
$\mathbb{E}_{F_0,G_0}(H_0)$
$\mathbb{E}_{F_0,G_1}(H_1)$
$\mathbb{E}_{F_1,G_0}(H_0)$
$\mathbb{E}_{F_1,G_1}(H_0)$

$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{E_1,H_1}(I_1)$

Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate

Garbled circuit framework [Yao86]



$\mathbb{E}_{A_0,B_0}(E_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$
$\mathbb{E}_{A_1,B_1}(E_0)$

$\mathbb{E}_{A_0,B_0}(F_0)$
$\mathbb{E}_{A_0,B_1}(F_1)$
$\mathbb{E}_{A_1,B_0}(F_1)$
$\mathbb{E}_{A_1,B_1}(F_0)$

$\mathbb{E}_{C_0,D_0}(G_0)$
$\mathbb{E}_{C_0,D_1}(G_1)$
$\mathbb{E}_{C_1,D_0}(G_0)$
$\mathbb{E}_{C_1,D_1}(G_0)$

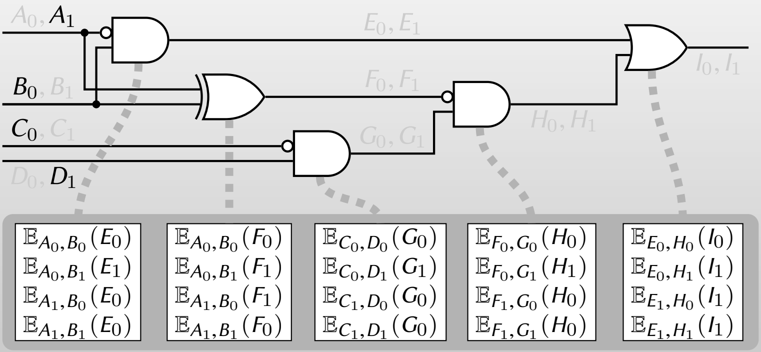
$\mathbb{E}_{F_0,G_0}(H_0)$
$\mathbb{E}_{F_0,G_1}(H_1)$
$\mathbb{E}_{F_1,G_0}(H_0)$
$\mathbb{E}_{F_1,G_1}(H_0)$

$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{E_1,H_1}(I_1)$

Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- "Encrypt" truth table of each gate
- **Garbled circuit** ≡ all encrypted gates

Garbled circuit framework [Yao86]

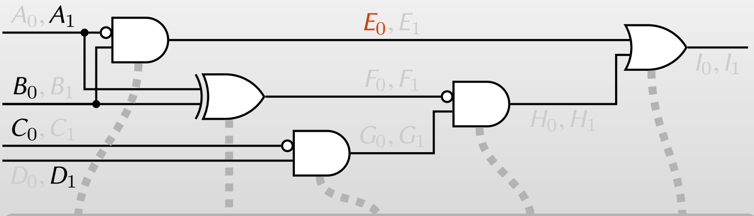


$\mathbb{E}_{A_0,B_0}(E_0)$	$\mathbb{E}_{A_0,B_0}(F_0)$	$\mathbb{E}_{C_0,D_0}(G_0)$	$\mathbb{E}_{F_0,G_0}(H_0)$	$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$	$\mathbb{E}_{A_0,B_1}(F_1)$	$\mathbb{E}_{C_0,D_1}(G_1)$	$\mathbb{E}_{F_0,G_1}(H_1)$	$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$	$\mathbb{E}_{A_1,B_0}(F_1)$	$\mathbb{E}_{C_1,D_0}(G_0)$	$\mathbb{E}_{F_1,G_0}(H_0)$	$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{A_1,B_1}(E_0)$	$\mathbb{E}_{A_1,B_1}(F_0)$	$\mathbb{E}_{C_1,D_1}(G_0)$	$\mathbb{E}_{F_1,G_1}(H_0)$	$\mathbb{E}_{E_1,H_1}(I_1)$

Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled circuit framework [Yao86]



$\mathbb{E}_{A_0,B_0}(E_0)$	$\mathbb{E}_{A_0,B_0}(F_0)$	$\mathbb{E}_{C_0,D_0}(G_0)$	$\mathbb{E}_{F_0,G_0}(H_0)$	$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$	$\mathbb{E}_{A_0,B_1}(F_1)$	$\mathbb{E}_{C_0,D_1}(G_1)$	$\mathbb{E}_{F_0,G_1}(H_1)$	$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$	$\mathbb{E}_{A_1,B_0}(F_1)$	$\mathbb{E}_{C_1,D_0}(G_0)$	$\mathbb{E}_{F_1,G_0}(H_0)$	$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{A_1,B_1}(E_0)$	$\mathbb{E}_{A_1,B_1}(F_0)$	$\mathbb{E}_{C_1,D_1}(G_0)$	$\mathbb{E}_{F_1,G_1}(H_0)$	$\mathbb{E}_{E_1,H_1}(I_1)$

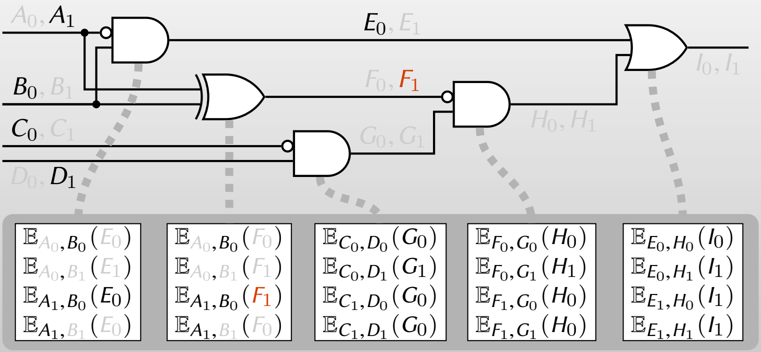
Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled evaluation:

- Only one ciphertext per gate is decryptable

Garbled circuit framework [Yao86]



$\mathbb{E}_{A_0,B_0}(E_0)$	$\mathbb{E}_{A_0,B_0}(F_0)$	$\mathbb{E}_{C_0,D_0}(G_0)$	$\mathbb{E}_{F_0,G_0}(H_0)$	$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$	$\mathbb{E}_{A_0,B_1}(F_1)$	$\mathbb{E}_{C_0,D_1}(G_1)$	$\mathbb{E}_{F_0,G_1}(H_1)$	$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$	$\mathbb{E}_{A_1,B_0}(F_1)$	$\mathbb{E}_{C_1,D_0}(G_0)$	$\mathbb{E}_{F_1,G_0}(H_0)$	$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{A_1,B_1}(E_0)$	$\mathbb{E}_{A_1,B_1}(F_0)$	$\mathbb{E}_{C_1,D_1}(G_0)$	$\mathbb{E}_{F_1,G_1}(H_0)$	$\mathbb{E}_{E_1,H_1}(I_1)$

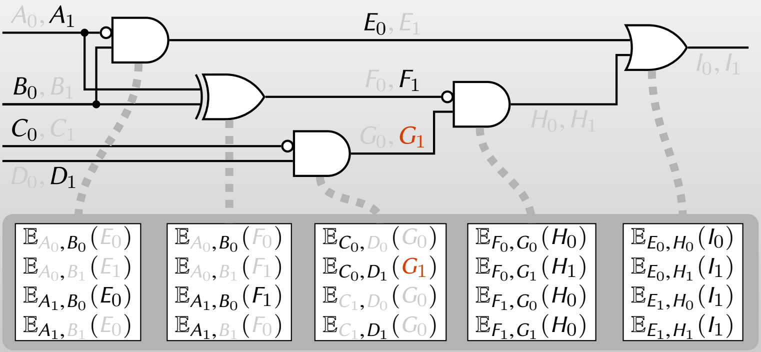
Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- "Encrypt" truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled evaluation:

- Only one ciphertext per gate is decryptable
- Result of decryption = value on outgoing wire

Garbled circuit framework [Yao86]



$\mathbb{E}_{A_0,B_0}(E_0)$	$\mathbb{E}_{A_0,B_0}(F_0)$	$\mathbb{E}_{C_0,D_0}(G_0)$	$\mathbb{E}_{F_0,G_0}(H_0)$	$\mathbb{E}_{E_0,H_0}(I_0)$
$\mathbb{E}_{A_0,B_1}(E_1)$	$\mathbb{E}_{A_0,B_1}(F_1)$	$\mathbb{E}_{C_0,D_1}(G_1)$	$\mathbb{E}_{F_0,G_1}(H_1)$	$\mathbb{E}_{E_0,H_1}(I_1)$
$\mathbb{E}_{A_1,B_0}(E_0)$	$\mathbb{E}_{A_1,B_0}(F_1)$	$\mathbb{E}_{C_1,D_0}(G_0)$	$\mathbb{E}_{F_1,G_0}(H_0)$	$\mathbb{E}_{E_1,H_0}(I_1)$
$\mathbb{E}_{A_1,B_1}(E_0)$	$\mathbb{E}_{A_1,B_1}(F_0)$	$\mathbb{E}_{C_1,D_1}(G_0)$	$\mathbb{E}_{F_1,G_1}(H_0)$	$\mathbb{E}_{E_1,H_1}(I_1)$

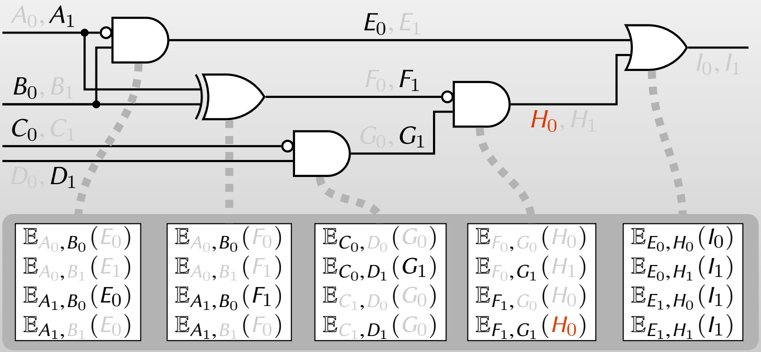
Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled evaluation:

- Only one ciphertext per gate is decryptable
- Result of decryption = value on outgoing wire

Garbled circuit framework [Yao86]



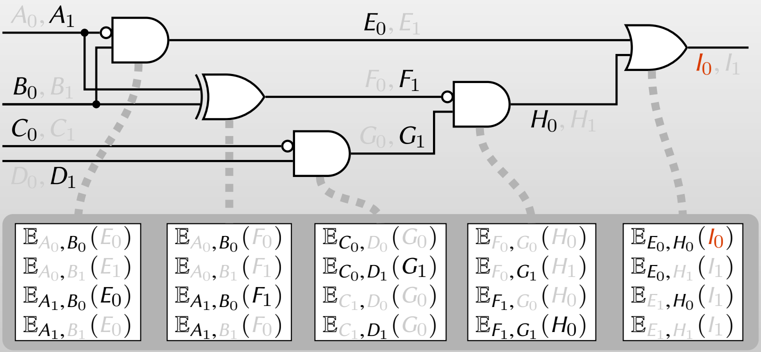
Garbling a circuit:

- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled evaluation:

- Only one ciphertext per gate is decryptable
- Result of decryption = value on outgoing wire

Garbled circuit framework [Yao86]



Garbling a circuit:

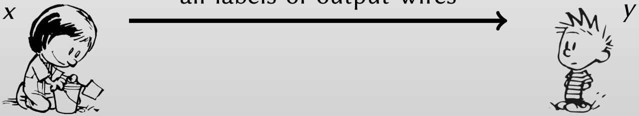
- Pick random **labels** W_0, W_1 on each wire
- “Encrypt” truth table of each gate
- **Garbled circuit** ≡ all encrypted gates
- **Garbled encoding** ≡ one label per wire

Garbled evaluation:

- Only one ciphertext per gate is decryptable
- Result of decryption = value on outgoing wire

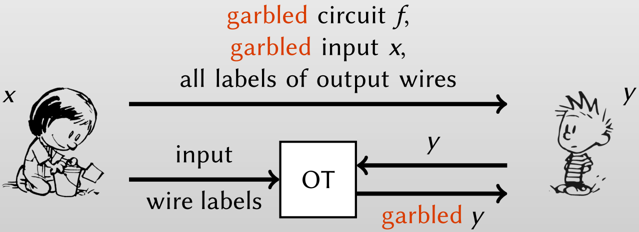
Applications: 2PC and more

Applications: 2PC and more



garbled circuit f,
garbled input x,
all labels of output wires

Applications: 2PC and more



x

garbled circuit f,
garbled input x,
all labels of output wires

input
wire labels

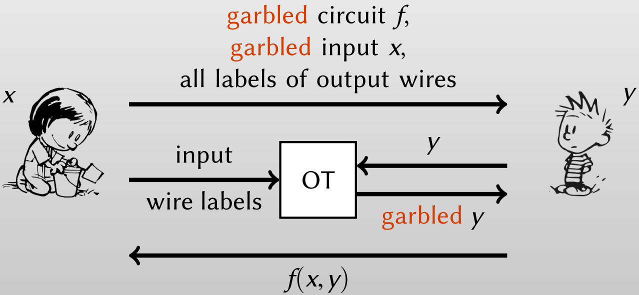
OT

y

garbled y

y

Applications: 2PC and more



x

garbled circuit f,
garbled input x,
all labels of output wires

y

input
wire labels

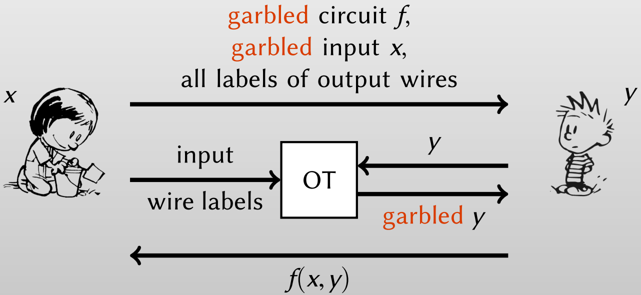
OT

y

garbled y

$f(x, y)$

Applications: 2PC and more


garbled circuit f,
garbled input x,
all labels of output wires

x

y

input wire labels

OT

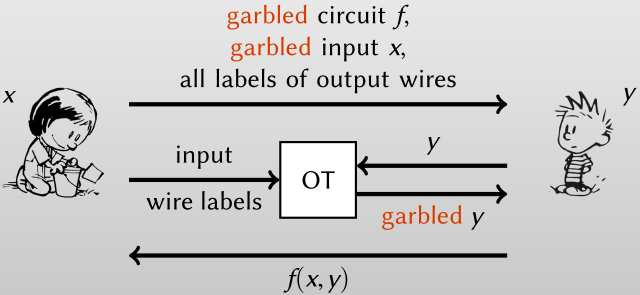
y

garbled y

$f(x, y)$

In 2PC: Parties agree on f to evaluate $\Rightarrow$ garbling doesn't have to hide f

Applications: 2PC and more



garbled circuit f,
garbled input x,
all labels of output wires

x

y

input
wire labels

OT

y

garbled y

$f(x, y)$

In 2PC: Parties agree on f to evaluate ⇒ garbling doesn't have to hide f

In other applications of garbled circuits it is helpful to **hide** f.

Gate-Hiding Garbled Circuits

Garbled circuit f + garbled input x

reveals no more than

$f(x)$ + topology of f

In particular, garbling **hides**:

- Values on non-output wires of f (including inputs x)
- Type of each gate (AND, OR, XOR, etc).

Garbled circuits: state of the art

	size ($\times\lambda$)		garble cost		eval cost		assump.
	XOR	AND	XOR	AND	XOR	AND	
Textbook Yao [Yao86,BMR90]	4		4		1		PRF
GRR3 [NPS99]	3		4		1		PRF
Free XOR [KS08]	0	3	0	4	0	1	circ+RK
GRR2 [PSSW09]	2	2	4	4	1	1	PRF
Half-gates [ZRE15]	0	2	4	4	2	2	circ+RK

Garbled circuits: state of the art

	size ($\times\lambda$)		garble cost		eval cost		assump.	gate hiding?
	XOR	AND	XOR	AND	XOR	AND		
Textbook Yao [Yao86,BMR90]	4		4		1		PRF	yes
GRR3 [NPS99]	3		4		1		PRF	yes
Free XOR [KS08]	0	3	0	4	0	1	circ+RK	**no**
GRR2 [PSSW09]	2	2	4	4	1	1	PRF	**no**
Half-gates [ZRE15]	0	2	4	4	2	2	circ+RK	**no**

Garbled circuits: state of the art

	size (×λ)		garble cost		eval cost		assump.	gate hiding?
	XOR	AND	XOR	AND	XOR	AND		
Textbook Yao [Yao86,BMR90]	4		4		1		PRF	yes
GRR3 [NPS99]	3		4		1		PRF	yes
Free XOR [KS08]	0	3	0	4	0	1	circ+RK	**no**
GRR2 [PSSW09]	2	2	4	4	1	1	PRF	**no**
Half-gates [ZRE15]	0	2	4	4	2	2	circ+RK	**no**

"no" = evaluation procedure depends on type of gate (e.g., XOR, AND)

Gate-hiding state of the art

	size	garble cost		eval cost		assump.
	$(\times\lambda)$	H	interp	H	interp	
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF
GRR3 [NPS99]	3	4	0	1	0	PRF

Gate-hiding state of the art

	size	garble cost		eval cost		assump.
	$(\times\lambda)$	H	interp	H	interp	
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF
GRR3 [NPS99]	3	4	0	1	0	PRF
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK

Gate-hiding state of the art

	size	garble cost		eval cost		assump.
	$(\times\lambda)$	H	interp	H	interp	
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF
GRR3 [NPS99]	3	4	0	1	0	PRF
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK
this paper #1	**2**	4	2	1	1	**PRF**
this paper #2	**2**	4	0	1	0	**PRF**

Gate-hiding state of the art

	size	garble cost		eval cost		assump.	gates
	$(\times\lambda)$	H	interp	H	interp		
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF	
GRR3 [NPS99]	3	4	0	1	0	PRF	
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK	
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK	
this paper #1	**2**	4	2	1	1	**PRF**	
this paper #2	**2**	4	0	1	0	**PRF**	

What kind of gates are actually supported?

Gate-hiding state of the art

	size	garble cost		eval cost		assump.	gates
	($\times\lambda$)	H	interp	H	interp		
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF	any
GRR3 [NPS99]	3	4	0	1	0	PRF	any
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK	
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK	
this paper #1	**2**	4	2	1	1	**PRF**	
this paper #2	**2**	4	0	1	0	**PRF**	

What kind of gates are actually supported?

Literally **any** gate
$g : \{0,1\}^2 \to \{0,1\}$

Gate-hiding state of the art

	size	garble cost		eval cost		assump.	gates
	$(\times\lambda)$	H	interp	H	interp		
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF	any
GRR3 [NPS99]	3	4	0	1	0	PRF	any
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK	symmetric
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK	symmetric
this paper #1	**2**	4	2	1	1	**PRF**	
this paper #2	**2**	4	0	1	0	**PRF**	

What kind of gates are actually supported?

Literally **any** gate
$g : \{0,1\}^2 \to \{0,1\}$

Symmetric gates
only: $g(1,0) = g(0,1)$

Gate-hiding state of the art

	size	garble cost		eval cost		assump.	gates
	$(\times\lambda)$	H	interp	H	interp		
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF	any
GRR3 [NPS99]	3	4	0	1	0	PRF	any
KKS [KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK	symmetric
WM [WangMalluhi17]	**2**	3	1	1	1	circ+RK	symmetric
this paper #1	**2**	4	2	1	1	**PRF**	non-const
this paper #2	**2**	4	0	1	0	**PRF**	non-const

What kind of gates are actually supported?

Literally **any** gate
$g : \{0,1\}^2 \to \{0,1\}$

Symmetric gates
only: $g(1,0) = g(0,1)$

All **except constant**
$g(a,b) = 0, g(a,b) = 1$

Our contribution

Two new garbled circuit constructions:

- **Gate-hiding**
- **Minimal size**
  2λ bits/gate matches state of the art for *standard* garbling
- **Minimal hardness assumption**: (PRF)
- **More natural class of gates**
  NOT gates can be absorbed into neighboring gates $\Rightarrow$ free

New Construction #1

	size ($\times\lambda$)	garble cost	eval cost	assump.	gate hiding?
GRR2 [PSSW09]	2	4	1	PRF	no

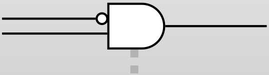
New Construction #1

	size ($\times\lambda$)	garble cost	eval cost	assump.	gate hiding?
GRR2 [PSSW09]	2	4	1	PRF	no ← why not?

New Construction #1

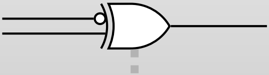
	size (×λ)	garble cost	eval cost	assump.	gate hiding?
GRR2 [PSSW09]	2	4	1	PRF	no ← why not?

Odd-parity gate:



0	0	0	
0	1	1	← odd # of 1s
1	0	0	
1	1	0	

Even-parity gate:

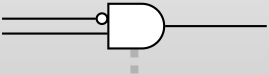


0	0	1	← even # of 1s
0	1	0	
1	0	0	
1	1	1	← even # of 1s

New Construction #1

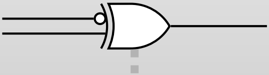
	size ($\times\lambda$)	garble cost	eval cost	assump.	gate hiding?
GRR2 [PSSW09]	2	4	1	PRF	no ← why not?

Odd-parity gate:



0	0	0	
0	1	1	← odd # of 1s
1	0	0	
1	1	0	

Even-parity gate:



0	0	1	← even # of 1s
0	1	0	
1	0	0	
1	1	1	← even # of 1s

[PinkasSchneiderSmartWilliams09]: **different techniques** for odd/even parity!

- **Our (simple) observation:** can adapt garbler method so that *odd-parity evaluation* works for even-parity gates too [details in backup slides]

New Construction #1

Garbled gate size: 2λ bits

Garbling cost:

- Finite field operations $\sim$ 2 interpolations of deg-2 polynomials
- 4 calls to cryptographic function $\mathbb{E}$

Evaluation cost:

- 1 interpolation of deg-2 polynomial
- 1 call to cryptographic function $\mathbb{E}$

Assumption: PRF

Gates supported: All except the two constant gates

New Construction #1

Garbled gate size: 2λ bits

Garbling cost:

- Finite field operations $\sim$ 2 interpolations of deg-2 polynomials
- 4 calls to cryptographic function $\mathbb{E}$

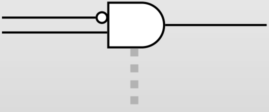
Evaluation cost:

- 1 interpolation of deg-2 polynomial
- 1 call to cryptographic function $\mathbb{E}$

Assumption: PRF

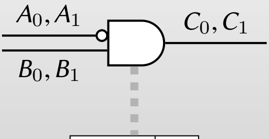
Gates supported: All except the two constant gates

Standard GC Tricks



0	0	0
0	1	1
1	0	0
1	1	0

Standard GC Tricks

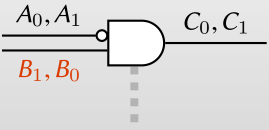


A_0	B_0	C_0
A_0	B_1	C_1
A_1	B_0	C_0
A_1	B_1	C_0

Decouple wire label subscript from **truth value**

- random association betwen $(0, 1) \leftrightarrow (T, F)$ on each wire

Standard GC Tricks

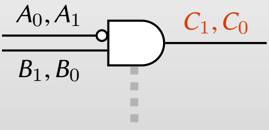


A_0	B_1	C_0
A_0	B_0	C_1
A_1	B_1	C_0
A_1	B_0	C_0

Decouple wire label subscript from **truth value**

- random association betwen $(0,1) \leftrightarrow (T,F)$ on each wire

Standard GC Tricks

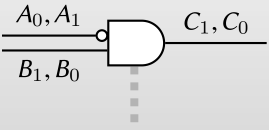


A_0	B_1	C_1
A_0	B_0	C_0
A_1	B_1	C_1
A_1	B_0	C_1

Decouple wire label subscript from **truth value**

- random association betwen $(0, 1) \leftrightarrow (T, F)$ on each wire

Standard GC Tricks



A_0	B_1	C_1
A_0	B_0	C_0
A_1	B_1	C_1
A_1	B_0	C_1

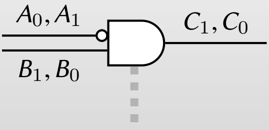
Decouple wire label subscript from **truth value**

- random association betwen $(0,1) \leftrightarrow (T,F)$ on each wire

Make wire label subscript **public** to evaluator

- e.g., least significant bit of label
- equivalent to including a “secret NOT gate”

Standard GC Tricks



A_0	B_1	C_1
A_0	B_0	C_0
A_1	B_1	C_1
A_1	B_0	C_1

Decouple wire label subscript from **truth value**

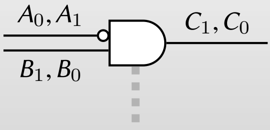
- random association betwen $(0, 1) \leftrightarrow (T, F)$ on each wire

Make wire label subscript **public** to evaluator

- e.g., least significant bit of label
- equivalent to including a “secret NOT gate”

⇒ Evaluator’s behavior can **depend on wire label subscripts** (“input combination”)

Standard GC Tricks



$K_1 = H(A_0, B_0)$	C_1
$K_2 = H(A_0, B_1)$	C_1
$K_3 = H(A_1, B_0)$	C_0
$K_4 = H(A_1, B_1)$	C_1

Decouple wire label subscript from **truth value**

- random association betwen $(0,1) \leftrightarrow (T, F)$ on each wire

Make wire label subscript **public** to evaluator

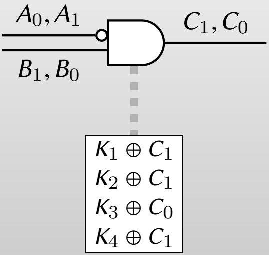
- e.g., least significant bit of label
- equivalent to including a "secret NOT gate"

⇒ Evaluator's behavior can **depend on wire label subscripts** ("input combination")

Use $K = H(A_i, B_j)$ as unique key for each input combination

- H can be built from a PRF in a simple way

Standard GC Tricks



Decouple wire label subscript from **truth value**

- random association betwen $(0,1) \leftrightarrow (T,F)$ on each wire

Make wire label subscript **public** to evaluator

- e.g., least significant bit of label
- equivalent to including a “secret NOT gate”

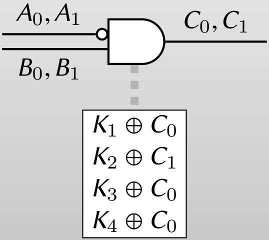
$\Rightarrow$ Evaluator’s behavior can **depend on wire label subscripts** (“input combination”)

Use $K = H(A_i, B_j)$ as unique key for each input combination

- H can be built from a PRF in a simple way

New Construction #2

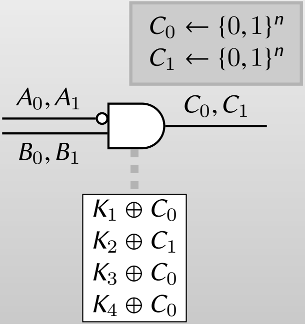
Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



$K_1 \oplus C_0$
$K_2 \oplus C_1$
$K_3 \oplus C_0$
$K_4 \oplus C_0$

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



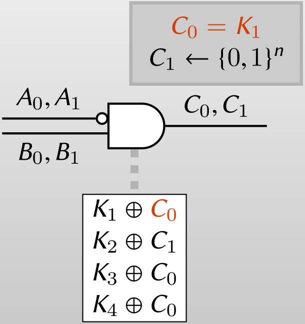
$C_0 \leftarrow \{0,1\}^n$
$C_1 \leftarrow \{0,1\}^n$

$K_1 \oplus C_0$
$K_2 \oplus C_1$
$K_3 \oplus C_0$
$K_4 \oplus C_0$

- Instead of choosing output wire labels randomly . . .

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



$C_0 = K_1$

$C_1 \leftarrow \{0,1\}^n$

A_0, A_1

B_0, B_1

C_0, C_1

$K_1 \oplus C_0$

$K_2 \oplus C_1$

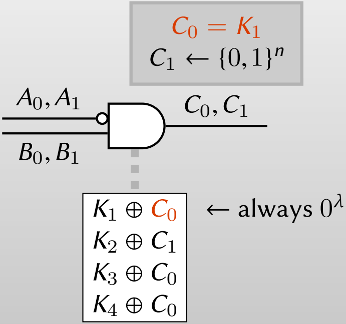
$K_3 \oplus C_0$

$K_4 \oplus C_0$

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



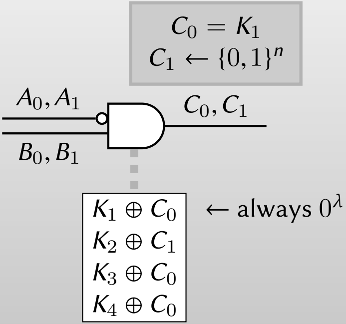
$C_0 = K_1$
$C_1 \leftarrow \{0,1\}^n$

$K_1 \oplus C_0$ ← always 0^λ
$K_2 \oplus C_1$
$K_3 \oplus C_0$
$K_4 \oplus C_0$

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



$$C_0 = K_1$$
$$C_1 \leftarrow \{0,1\}^n$$

$$\boxed{\begin{array}{c} K_1 \oplus C_0 \\ K_2 \oplus C_1 \\ K_3 \oplus C_0 \\ K_4 \oplus C_0 \end{array}}$$

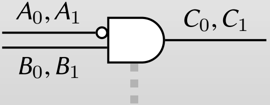
$\leftarrow$ always 0^λ

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:

$$C_0 = K_1$$
$$C_1 = K_2 \oplus K_3 \oplus K_4$$



$$\begin{array}{|c|} \hline K_1 \oplus C_0 \\ K_2 \oplus C_1 \\ K_3 \oplus C_0 \\ K_4 \oplus C_0 \\ \hline \end{array}$$

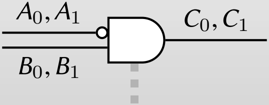
← always 0^λ

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:

$$\begin{aligned} C_0 &= K_1 \\ C_1 &= K_2 \oplus K_3 \oplus K_4 \end{aligned}$$



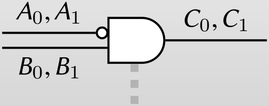
$$\begin{array}{|c|l} \hline K_1 \oplus C_0 & \leftarrow \text{always } 0^\lambda \\ K_2 \oplus C_1 & \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \\ K_3 \oplus C_0 & \quad \leftarrow \text{always xor to } 0^\lambda \\ K_4 \oplus C_0 & \\ \hline \end{array}$$

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:

$$\boxed{\begin{array}{r} C_0 = K_1 \\ C_1 = K_2 \oplus K_3 \oplus K_4 \end{array}}$$

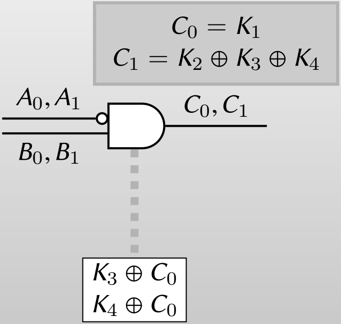


$$\boxed{\begin{array}{l} K_1 \oplus C_0 \\ K_2 \oplus C_1 \\ K_3 \oplus C_0 \\ K_4 \oplus C_0 \end{array}} \begin{array}{l} \leftarrow \text{always } 0^\lambda \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \leftarrow \text{always xor to } 0^\lambda \end{array}$$

- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero
- First 2 ciphertexts are **linear combination** of last 2

New Construction #2

Inspired by [GueronLindellNofPinkas15] technique **for odd-parity gates only**:



$$C_0 = K_1$$
$$C_1 = K_2 \oplus K_3 \oplus K_4$$

$$K_3 \oplus C_0$$
$$K_4 \oplus C_0$$

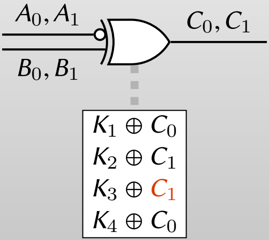
- Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero
- First 2 ciphertexts are **linear combination** of last 2 ⇒ don't send them! (evaulator can reconstruct first 2 "virtually")

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



A_0, A_1

B_0, B_1

C_0, C_1

$K_1 \oplus C_0$

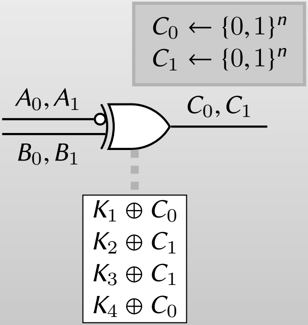
$K_2 \oplus C_1$

$K_3 \oplus C_1$

$K_4 \oplus C_0$

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



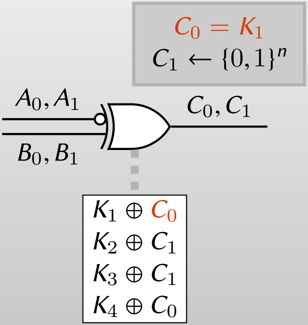
$$C_0 \leftarrow \{0,1\}^n$$
$$C_1 \leftarrow \{0,1\}^n$$

$$K_1 \oplus C_0$$
$$K_2 \oplus C_1$$
$$K_3 \oplus C_1$$
$$K_4 \oplus C_0$$

- (Same as before) Instead of choosing output wire labels randomly . . .

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



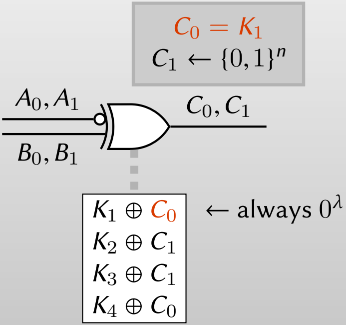
$$C_0 = K_1$$
$$C_1 \leftarrow \{0,1\}^n$$

$$\begin{array}{l} K_1 \oplus C_0 \\ K_2 \oplus C_1 \\ K_3 \oplus C_1 \\ K_4 \oplus C_0 \end{array}$$

- (Same as before) Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



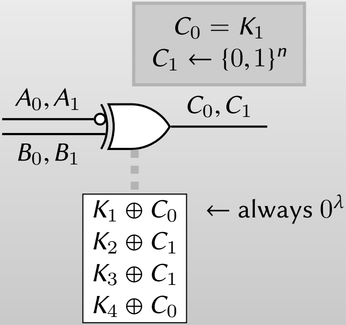
$C_0 = K_1$
$C_1 \leftarrow \{0,1\}^n$

$K_1 \oplus C_0$ ← always 0^λ
$K_2 \oplus C_1$
$K_3 \oplus C_1$
$K_4 \oplus C_0$

- (Same as before) Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



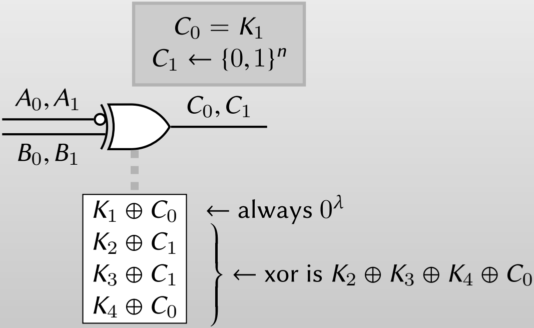
$C_0 = K_1$
$C_1 \leftarrow \{0,1\}^n$

$K_1 \oplus C_0$ ← always 0^λ
$K_2 \oplus C_1$
$K_3 \oplus C_1$
$K_4 \oplus C_0$

- (Same as before) Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero

New Construction #2

Why doesn't [GueronLindellNofPinkas15] doesn't work for **even-parity** gates?



$C_0 = K_1$

$C_1 \leftarrow \{0,1\}^n$

A_0, A_1

B_0, B_1

C_0, C_1

$$\begin{array}{|c|}\hline K_1 \oplus C_0 \\ K_2 \oplus C_1 \\ K_3 \oplus C_1 \\ K_4 \oplus C_0 \\ \hline \end{array}$$

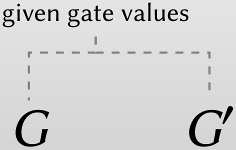
$\leftarrow$ always 0^λ

$\left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \leftarrow$ xor is $K_2 \oplus K_3 \oplus K_4 \oplus C_0$

- (Same as before) Instead of choosing output wire labels randomly . . .
- . . . choose them to make 1st ciphertext zero, and other 3 ciphertexts xor to zero ???
- But xor of other 3 ciphertexts already fixed! (C_1 cancels out!)

New Construction #2

Abstracting **evaluator's behavior** in [GueronLindellNofPinkas15]:



given gate values

G G'

From the two given values for this garbled gate . . .

New Construction #2

Abstracting **evaluator's behavior** in [GueronLindellNofPinkas15]:

$$\underbrace{\alpha_i \overbrace{G \oplus \beta_i G'}^{\text{given gate values}}}_{\text{virtual row}}$$

From the two given values for this garbled gate . . .

. . . reconstruct "virtual row" ciphertext as linear combination

New Construction #2

Abstracting **evaluator's behavior** in [GueronLindellNofPinkas15]:

given gate values

key derived from input labels

$$K_i \qquad \underbrace{\alpha_i\, G \oplus \beta_i\, G'}_{\text{virtual row}}$$

From the two given values for this garbled gate . . .

. . . reconstruct "virtual row" ciphertext as linear combination

Compute key unique to this input combination

New Construction #2

Abstracting **evaluator's behavior** in [GueronLindellNofPinkas15]:

given gate values

key derived from input labels

$$C := K_i \oplus \underbrace{\alpha_i \, G \oplus \beta_i \, G'}_{\text{virtual row}}$$

output label

From the two given values for this garbled gate . . .

. . . reconstruct "virtual row" ciphertext as linear combination

Compute key unique to this input combination and decrypt virtual row

New Construction #2

$$C := K_i \oplus \alpha_i \, G \oplus \beta_i \, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

New Construction #2

$$C := K_i \oplus \alpha_i \, G \oplus \beta_i \, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

New Construction #2

$$C := K_i \oplus \alpha_i \, G \oplus \beta_i \, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

Our idea:

- random & secret

New Construction #2

$$C := K_i \oplus \alpha_i \, G \oplus \beta_i \, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

Our idea:

- random & secret

Want evaluation to work like this:

New Construction #2

$$C := K_i \oplus \alpha_i\, G \oplus \beta_i\, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

Our idea:

- random & secret

Want evaluation to work like this:

- Garbled gate: G, G' plus **encryptions** of $(\alpha_1, \beta_1), \ldots, (\alpha_4, \beta_4)$

New Construction #2

$$C := K_i \oplus \alpha_i \, G \oplus \beta_i \, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

Our idea:

- random & secret

Want evaluation to work like this:

- Garbled gate: G, G' plus **encryptions** of $(\alpha_1, \beta_1), \ldots, (\alpha_4, \beta_4)$
- Evaluator can only decrypt appropriate α_i, β_i

New Construction #2

$$C := K_i \oplus \alpha_i\, G \oplus \beta_i\, G'$$

α_i, β_i coefficients are **bits** that depend on input combination.

[GueronLindellNofPinkas15]:

- fixed & public

Our idea:

- random & secret

Want evaluation to work like this:

- Garbled gate: G, G' plus **encryptions** of $(\alpha_1, \beta_1), \ldots, (\alpha_4, \beta_4)$
- Evaluator can only decrypt appropriate α_i, β_i
- Computes output label as $C := K_i \oplus \alpha_i G \oplus \beta_i G'$

New Construction #2

To have correctness, we need:

$$
\begin{aligned}
C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\
C_0 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\
C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\
C_0 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G'
\end{aligned}
$$

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_0 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_0 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 1 & 0 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 1 & 0 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_1 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_0 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \quad \iff \quad \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 0 & 1 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 1 & 0 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

(different gate types affect first two columns of matrix)

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_1 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_1 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \quad \Longleftrightarrow \quad \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 0 & 1 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 0 & 1 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

(different gate types affect first two columns of matrix)

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_1 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_1 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 0 & 1 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 0 & 1 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

(different gate types affect first two columns of matrix)

Key idea

Garbler samples (α_i, β_i) uniformly, **subject to matrix being invertible**, then solves for C_0, C_1, G, G' given $K_1, \ldots, K_4$

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_1 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_1 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \quad \iff \quad \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 0 & 1 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 0 & 1 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

(different gate types affect first two columns of matrix)

Key idea

Garbler samples (α_i, β_i) uniformly, **subject to matrix being invertible**, then solves for C_0, C_1, G, G' given $K_1, \ldots, K_4$

- Different gate types induce **different distribution** over (α_i, β_i) bits
- Evaluator sees only one **particular** (α_i, β_i) value (others encrypted)
- Different distributions have same **marginals** ⇒ hides gate type

New Construction #2

To have correctness, we need:

$$\begin{aligned} C_0 &= K_1 \oplus \alpha_1 G \oplus \beta_1 G' \\ C_1 &= K_2 \oplus \alpha_2 G \oplus \beta_2 G' \\ C_1 &= K_3 \oplus \alpha_3 G \oplus \beta_3 G' \\ C_1 &= K_4 \oplus \alpha_4 G \oplus \beta_4 G' \end{aligned} \quad \Longleftrightarrow \quad \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 0 & 1 & \alpha_2 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 0 & 1 & \alpha_4 & \beta_4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ G \\ G' \end{bmatrix}$$

(different gate types affect first two columns of matrix)

Key idea

Garbler samples (α_i, β_i) uniformly, **subject to matrix being invertible**, then solves for C_0, C_1, G, G' given $K_1, \ldots, K_4$

- Different gate types induce **different distribution** over (α_i, β_i) bits
- Evaluator sees only one **particular** (α_i, β_i) value (others encrypted)
- Different distributions have same **marginals** $\Rightarrow$ hides gate type
- *(matrix invertible unless this is a constant gate)*

New Construction #2

Garbled gate size: 2λ bits, plus 8 bits to encrypt α_i, β_i values

Garbling cost:

- 4 calls to cryptographic function $\mathbb{E}$
- no finite field operations (just xor)

Evaluation cost:

- 1 call to cryptographic function $\mathbb{E}$
- no finite field operations (just xor)

Assumption: PRF

Gates supported: All except the two constant gates

Summary

Two new garbled circuit constructions:

- **Gate-hiding**
- **Minimal size** (2λ bits/gate)
- **Minimal hardness assumption**: (PRF)
- **More natural class of gates** (all gates except two constant gates)

	size	garble cost		eval cost		assump.	gates
	($\times\lambda$)	H	interp	H	interp		
Textbook [Yao86,BMR90]	4	4	0	1	0	PRF	any
GRR3 [NPS99]	3	4	0	1	0	PRF	any
[KempkaKikuchiSuziki16]	**2**	3	0	1	0	circ+RK	symm
[WangMalluhi17]	**2**	3	1	1	1	circ+RK	symm
this paper #1	**2**	4	2	1	1	**PRF**	non-const
this paper #2	**2**	4	0	1	0	**PRF**	non-const

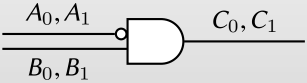
the end.

any questions?

New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:



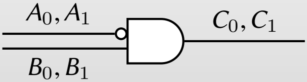
New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0)$$
$$K_2 = H(A_0, B_1)$$
$$K_3 = H(A_1, B_0)$$
$$K_4 = H(A_1, B_1)$$



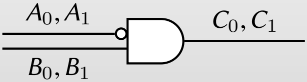
New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$



New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

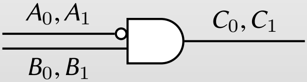
Idea: Evaluator can know exactly one of:

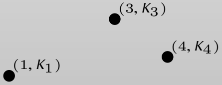
$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$

$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$





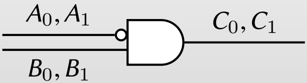
$$(1, K_1), (3, K_3), (4, K_4)$$

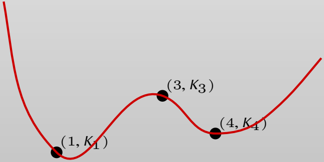
New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

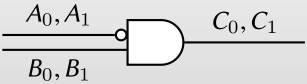
Idea: Evaluator can know exactly one of:

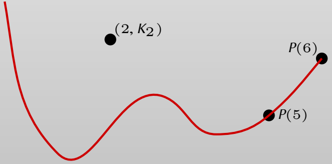
$K_1 = H(A_0, B_0) \rightsquigarrow$ learn C_0

$K_2 = H(A_0, B_1) \rightsquigarrow$ learn C_1

$K_3 = H(A_1, B_0) \rightsquigarrow$ learn C_0

$K_4 = H(A_1, B_1) \rightsquigarrow$ learn C_0





$P =$ uniq deg-2 poly thru

$(1, K_1), (3, K_3), (4, K_4)$

$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

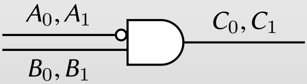
Idea: Evaluator can know exactly one of:

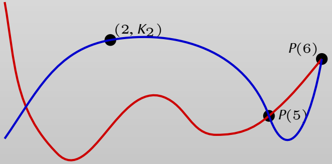
$K_1 = H(A_0, B_0) \rightsquigarrow$ learn C_0

$K_2 = H(A_0, B_1) \rightsquigarrow$ learn C_1

$K_3 = H(A_1, B_0) \rightsquigarrow$ learn C_0

$K_4 = H(A_1, B_1) \rightsquigarrow$ learn C_0





$P =$ uniq deg-2 poly thru

$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru

$(2, K_2), (5, P(5)), (6, P(6))$

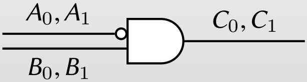
New Construction #1

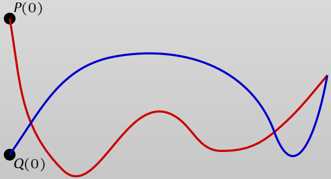
Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

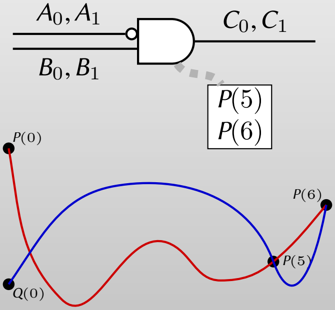
New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$



$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$
\begin{aligned}
K_1 &= H(A_0, B_0) \rightsquigarrow \text{learn } C_0\\
K_2 &= H(A_0, B_1) \rightsquigarrow \text{learn } C_1\\
K_3 &= H(A_1, B_0) \rightsquigarrow \text{learn } C_0\\
K_4 &= H(A_1, B_1) \rightsquigarrow \text{learn } C_0
\end{aligned}
$$

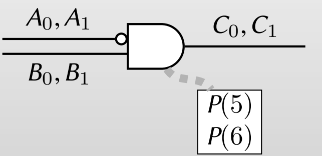
To evaluate a gate:

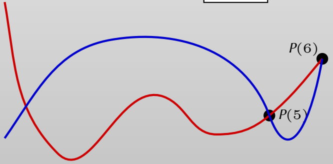
- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

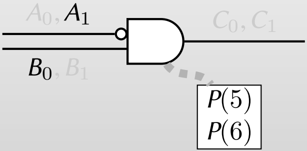
To evaluate a gate:

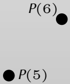
- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$

$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

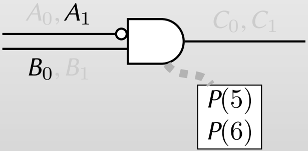
To evaluate a gate:

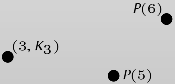
- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$

$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

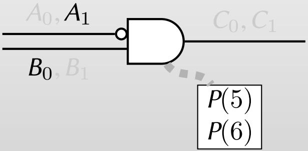
To evaluate a gate:

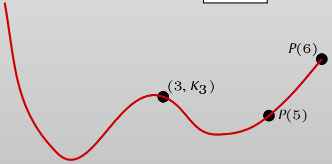
- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$





$P =$ uniq deg-2 poly thru $(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru $(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:

[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$

$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

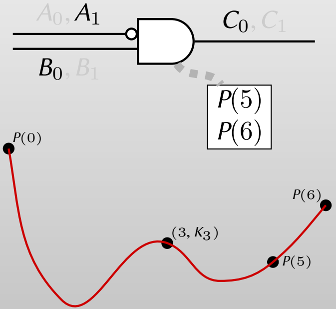
To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$$\boxed{C_0 = P(0); C_1 = Q(0)}$$



$P =$ uniq deg-2 poly thru $(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru $(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$K_1 = H(A_0, B_0) \rightsquigarrow$ learn C_0

$K_2 = H(A_0, B_1) \rightsquigarrow$ learn C_1

$K_3 = H(A_1, B_0) \rightsquigarrow$ learn C_0

$K_4 = H(A_1, B_1) \rightsquigarrow$ learn C_0

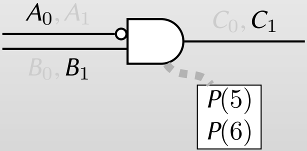
To evaluate a gate:

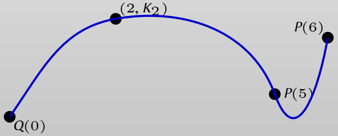
- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$C_0 = P(0); C_1 = Q(0)$





$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

Starting point:
[PinkasSchneiderSmartWilliams09]:

Idea: Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

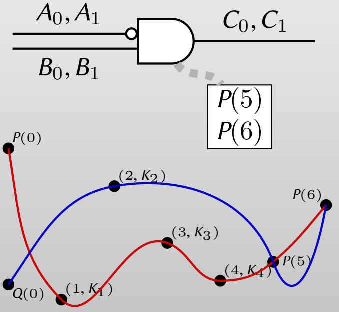
To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

$C_0 = P(0); C_1 = Q(0)$



$P =$ uniq deg-2 poly thru
$(1, K_1), (3, K_3), (4, K_4)$

$Q =$ uniq deg-2 poly thru
$(2, K_2), (5, P(5)), (6, P(6))$

New Construction #1

[PinkasSchneiderSmartWilliams09] technique works for any **odd-parity** gate

- odd # of 1s in the truth table (e.g., AND, NOR)
- 3 of (i, K_i) on 1 polynomial, other (i, K_i) on another polynomial

New Construction #1

[PinkasSchneiderSmartWilliams09] technique works for any **odd-parity** gate

- odd # of 1s in the truth table (e.g., AND, NOR)
- 3 of (i, K_i) on 1 polynomial, other (i, K_i) on another polynomial

They show a **different technique** for even-parity gates (e.g., XOR, XNOR)

New Construction #1

[PinkasSchneiderSmartWilliams09] technique works for any **odd-parity** gate

- odd # of 1s in the truth table (e.g., AND, NOR)
- 3 of (i, K_i) on 1 polynomial, other (i, K_i) on another polynomial

They show a **different technique** for even-parity gates (e.g., XOR, XNOR)

Our contribution:

Can make odd-parity evaluation procedure work for even parity gates too!

New Construction #1

Same as before

Evaluator can know exactly one of:

$$\begin{aligned} K_1 &= H(A_0, B_0) \\ K_2 &= H(A_0, B_1) \\ K_3 &= H(A_1, B_0) \\ K_4 &= H(A_1, B_1) \end{aligned}$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

New Construction #1

Same as before, but **even-parity** gate:

Evaluator can know exactly one of:

$$\begin{aligned}
K_1 &= H(A_0, B_0) \rightsquigarrow \text{learn } C_0 \\
K_2 &= H(A_0, B_1) \rightsquigarrow \text{learn } C_1 \\
K_3 &= H(A_1, B_0) \rightsquigarrow \text{learn } C_1 \\
K_4 &= H(A_1, B_1) \rightsquigarrow \text{learn } C_0
\end{aligned}$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

  $$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

New Construction #1

Same as before, but **even-parity** gate:

Evaluator can know exactly one of:

$$\begin{aligned}
K_1 &= H(A_0, B_0) \rightsquigarrow \text{learn } C_0 \\
K_2 &= H(A_0, B_1) \rightsquigarrow \text{learn } C_1 \\
K_3 &= H(A_1, B_0) \rightsquigarrow \text{learn } C_1 \\
K_4 &= H(A_1, B_1) \rightsquigarrow \text{learn } C_0
\end{aligned}$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

Need:

- deg-2 polynomials P & Q

$$\begin{bmatrix} \\ \\ \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix} & & & & & \\ \\ \\ \\ \\ \\ \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

New Construction #1

Same as before, but **even-parity** gate:

Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_1$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

Need:

- deg-2 polynomials P & Q
- $P(5) = Q(5)$; $P(6) = Q(6)$

$$\begin{bmatrix} \\ \\ \\ \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

New Construction #1

Same as before, but **even-parity** gate:

Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$

$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$

$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_1$$

$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

Need:

- deg-2 polynomials P & Q
- $P(5) = Q(5)$; $P(6) = Q(6)$
- P goes through $(1, K_1), (4, K_4)$

$$\begin{bmatrix} K_1 \\ \\ \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1^0 & 1^1 & 1^2 & & & \\ & & & & & \\ & & & & & \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

New Construction #1

Same as before, but **even-parity** gate:

Evaluator can know exactly one of:

$$K_1 = H(A_0, B_0) \rightsquigarrow \text{learn } C_0$$
$$K_2 = H(A_0, B_1) \rightsquigarrow \text{learn } C_1$$
$$K_3 = H(A_1, B_0) \rightsquigarrow \text{learn } C_1$$
$$K_4 = H(A_1, B_1) \rightsquigarrow \text{learn } C_0$$

To evaluate a gate:

- Compute relevant K_i & interpolate:

$$(i, K_i), (5, P(5)), (6, P(6))$$

- Evaluate polynomial at zero

Need:

- deg-2 polynomials P & Q
- $P(5) = Q(5)$; $P(6) = Q(6)$
- P goes through $(1, K_1), (4, K_4)$
- Q goes through $(2, K_2), (3, K_3)$

$$\begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1^0 & 1^1 & 1^2 & & & \\ & & & 2^0 & 2^1 & 2^2 \\ & & & 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

New Construction #1

What we need (for evaluation to work):

$$\left\{\begin{array}{rcl} P(1) & = & K_1 \\ Q(2) & = & K_2 \\ Q(3) & = & K_3 \\ P(4) & = & K_4 \\ P(5)-Q(5) & = & 0 \\ P(6)-Q(6) & = & 0 \end{array}\right\} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1^0 & 1^1 & 1^2 & & & \\ & & & 2^0 & 2^1 & 2^2 \\ & & & 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

New Construction #1

What we need (for evaluation to work):

$$\left\{\begin{aligned} P(1) &= K_1 \\ Q(2) &= K_2 \\ Q(3) &= K_3 \\ P(4) &= K_4 \\ P(5) - Q(5) &= 0 \\ P(6) - Q(6) &= 0 \end{aligned}\right\} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1^0 & 1^1 & 1^2 & & & \\ & & & 2^0 & 2^1 & 2^2 \\ & & & 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

Garbling procedure:

- Compute $K_1, \ldots, K_4$ (depend on incoming wire labels)
- Invert this matrix to solve for polynomials P and Q
- Garbled gate is $(P(5), Q(5))$

New Construction #1

What we need (for evaluation to work):

$$\left\{\begin{aligned} P(1) &= K_1 \\ Q(2) &= K_2 \\ Q(3) &= K_3 \\ P(4) &= K_4 \\ P(5) - Q(5) &= 0 \\ P(6) - Q(6) &= 0 \end{aligned}\right\} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1^0 & 1^1 & 1^2 & & & \\ & & & 2^0 & 2^1 & 2^2 \\ & & & 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

Garbling procedure:

- Compute $K_1, \ldots, K_4$ (depend on incoming wire labels)
- Invert this matrix to solve for polynomials P and Q
- Garbled gate is $(P(5), Q(5))$

Main observation: this matrix is invertible for any non-constant gate

New Construction #1

What we need (for evaluation to work):

$$\left\{\begin{aligned} Q(1) &= K_1 \\ Q(2) &= K_2 \\ Q(3) &= K_3 \\ P(4) &= K_4 \\ P(5)-Q(5) &= 0 \\ P(6)-Q(6) &= 0 \end{aligned}\right\} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} & & & 1^0 & 1^1 & 1^2 \\ & & & 2^0 & 2^1 & 2^2 \\ & & & 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

Garbling procedure:

- Compute $K_1, \ldots, K_4$ (depend on incoming wire labels)
- Invert this matrix to solve for polynomials P and Q
- Garbled gate is $(P(5), Q(5))$

Main observation: this matrix is invertible for any non-constant gate

New Construction #1

What we need (for evaluation to work):

$$\left\{\begin{array}{rcl} Q(1) &=& K_1 \\ Q(2) &=& K_2 \\ P(3) &=& K_3 \\ P(4) &=& K_4 \\ P(5)-Q(5) &=& 0 \\ P(6)-Q(6) &=& 0 \end{array}\right\} \iff \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} & & & 1^0 & 1^1 & 1^2 \\ & & & 2^0 & 2^1 & 2^2 \\ 3^0 & 3^1 & 3^2 & & & \\ 4^0 & 4^1 & 4^2 & & & \\ 5^0 & 5^1 & 5^2 & -5^0 & -5^1 & -5^2 \\ 6^0 & 6^1 & 6^2 & -6^0 & -6^1 & -6^2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ q_0 \\ q_1 \\ q_2 \end{bmatrix}$$

Garbling procedure:

- Compute $K_1, \ldots, K_4$ (depend on incoming wire labels)
- Invert this matrix to solve for polynomials P and Q
- Garbled gate is $(P(5), Q(5))$

Main observation: this matrix is invertible for any non-constant gate