State of the art techniques

Lindell13: s circuits + aux comp

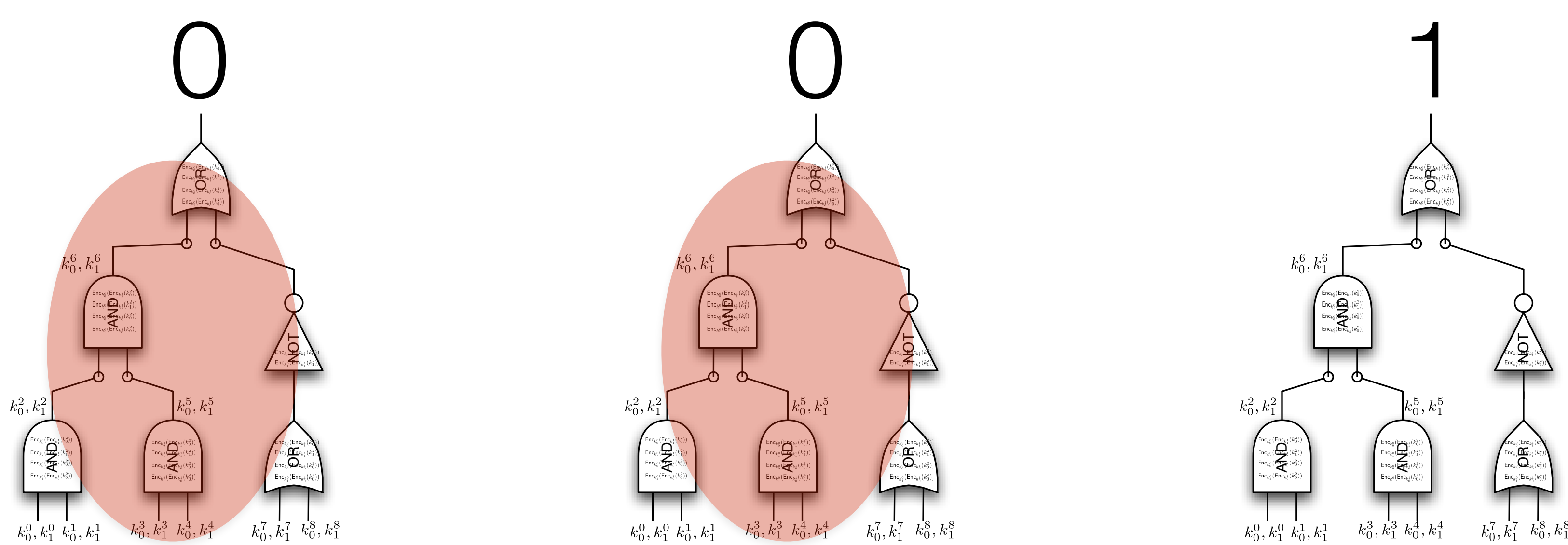
HKE13: 2s circuits + aux comp

Rethinking circuit consistency

$$\min_t \max_c \frac{\binom{k-c}{t}}{\binom{k}{t}}$$

Bottleneck is requirement for majority good circuits.

Avoiding majority

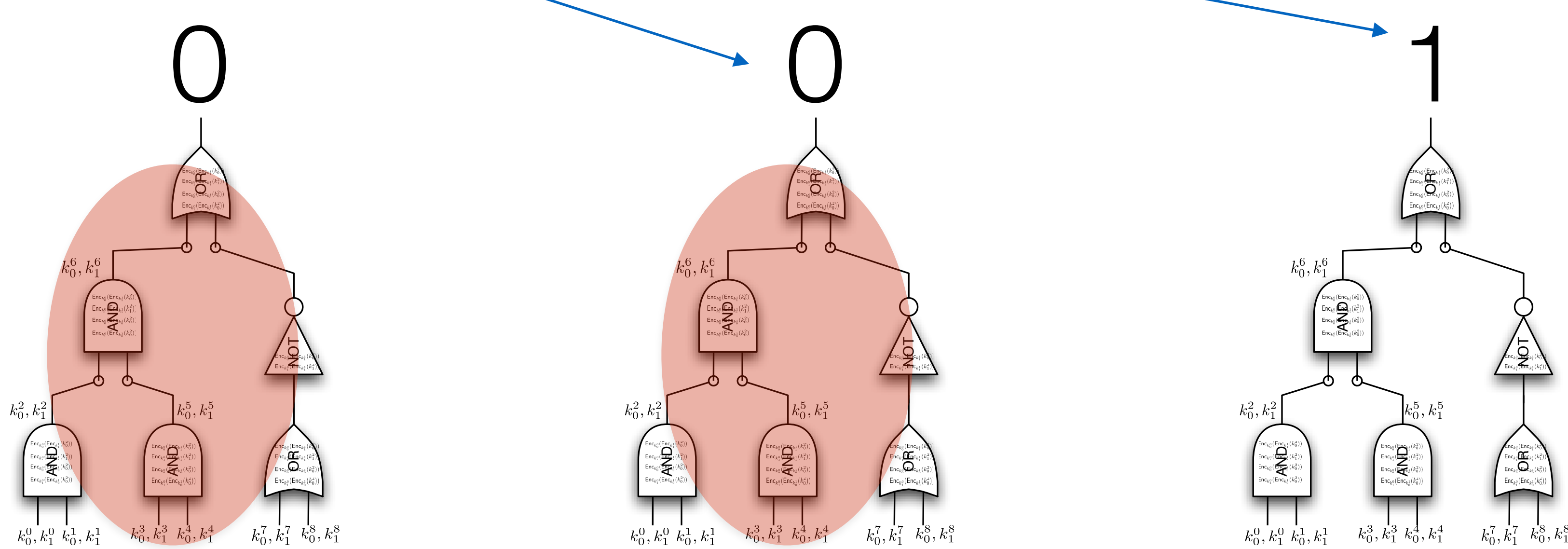


Different values for the same output wire imply cheating.

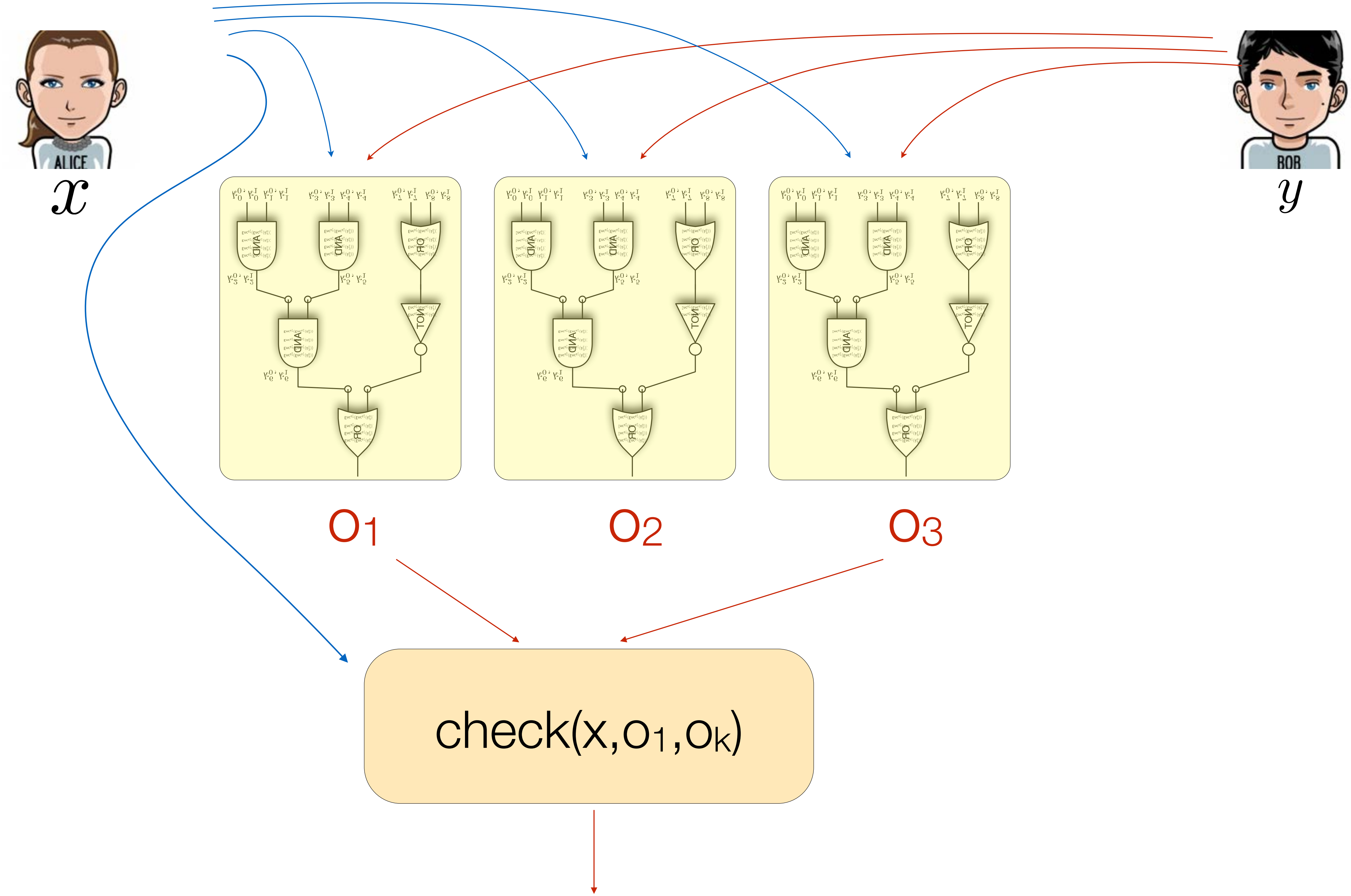
By previous attack, Evaluator cannot acknowledge cheating!

Key idea

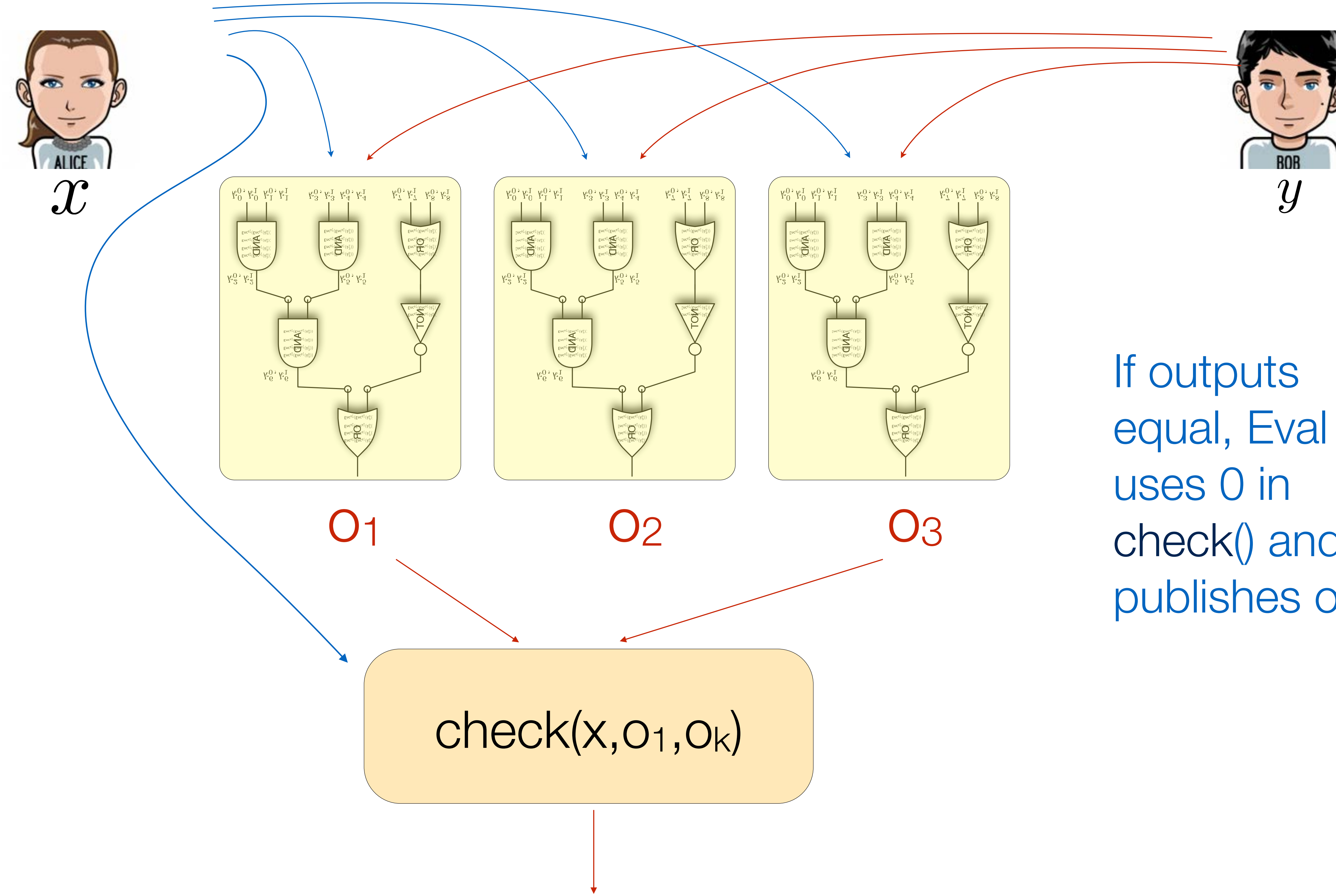
witness of cheating: 0+1
label for same output bit.



Use the “witness of cheating”
to unlock Garbler’s input.

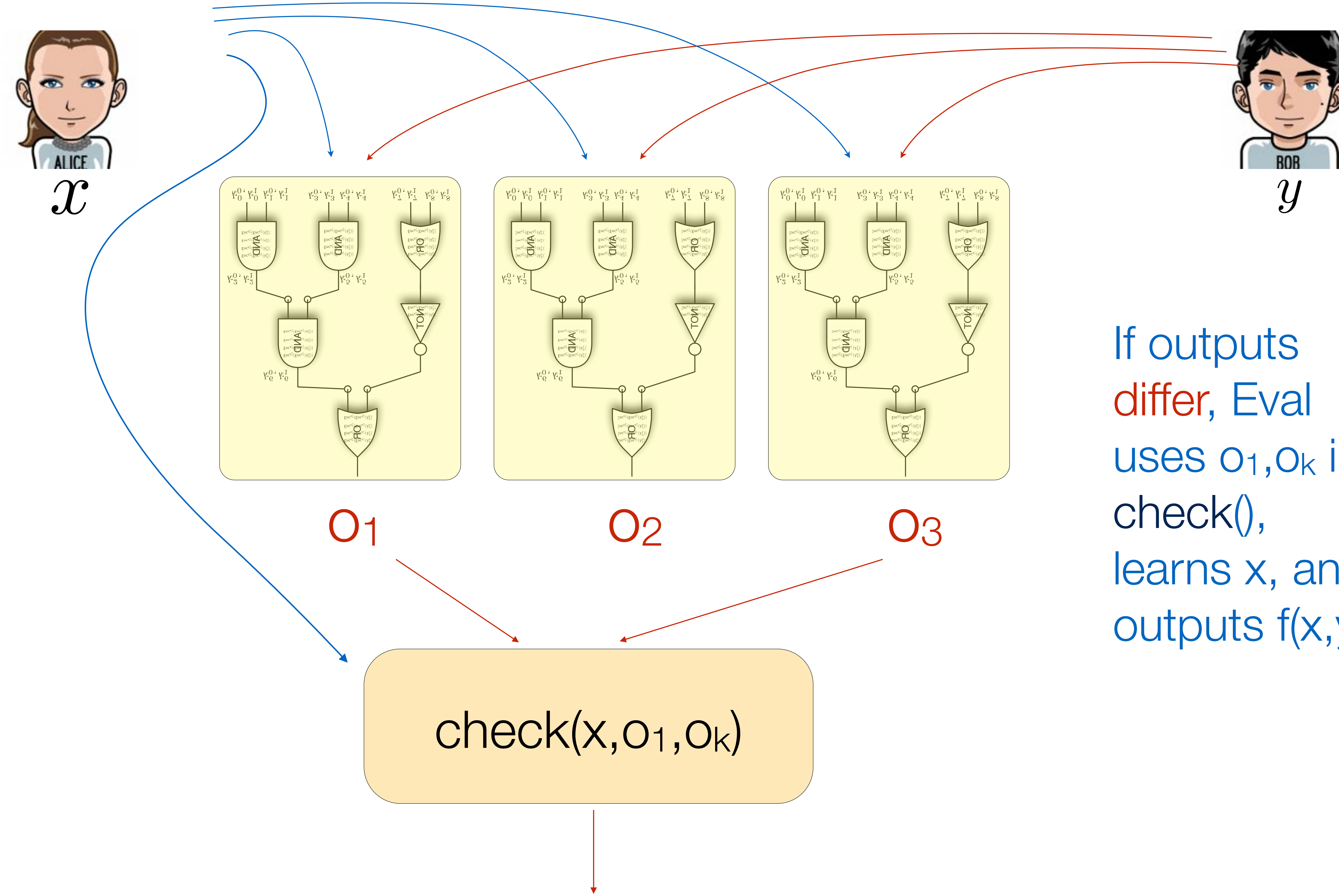


x iff o_1,o_k are “valid” output labels for 0,1



If outputs equal, Eval uses 0 in check() and publishes o_1

x iff o_1,o_k are “valid” output labels for 0,1

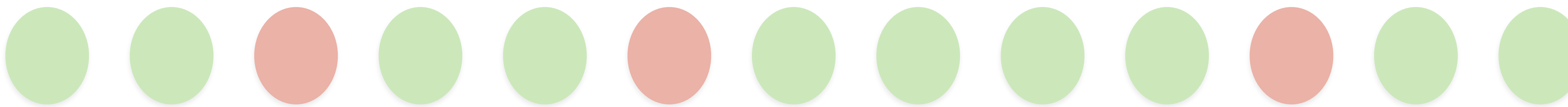


If outputs differ, Eval uses o_1,o_k in check(), learns x, and outputs f(x,y)

x iff o_1,o_k are “valid” output labels for 0,1

New Analysis

k circuits in total



Garbler picks a subset C^* of circuits to corrupt.

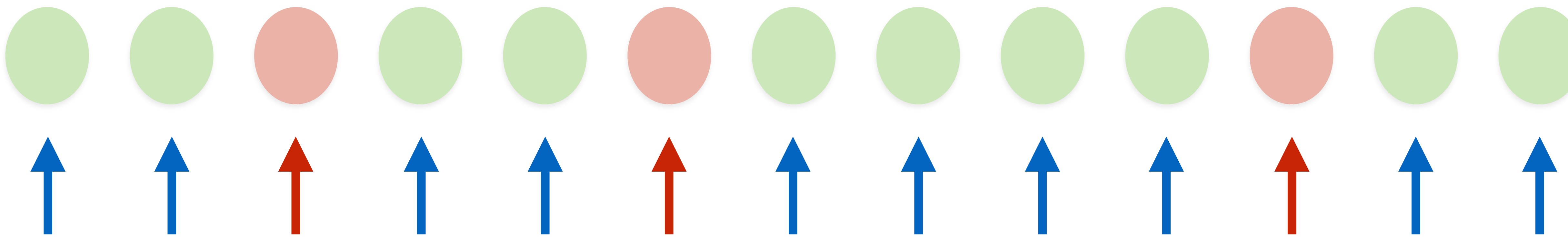
Evaluator picks a subset T^* of circuits to test.

Choice of T^* is uniformly random over all subsets.

Let G^* be the set of good circuits. $G^* = \overline{C^*}$.

Suppose G* = T*

k circuits in total

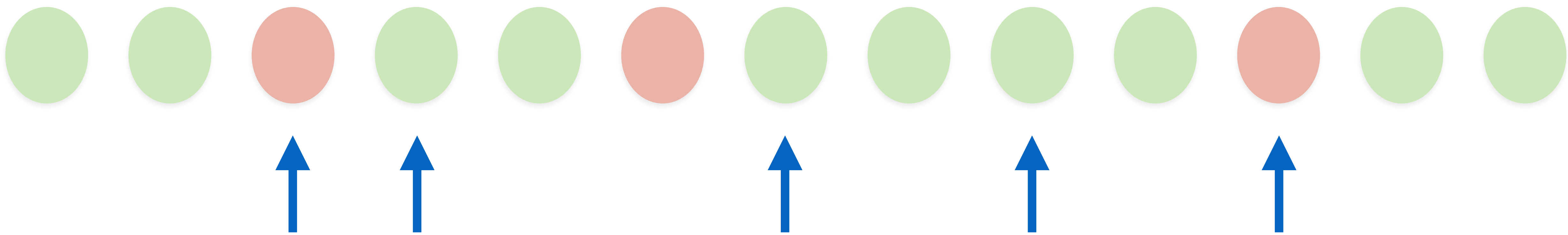


Garbler picks a subset C* of circuits to corrupt.

Choice of T* is uniformly random over all subsets.

Failure, but $\Pr[G^* = T^*] = 2^{-k}$

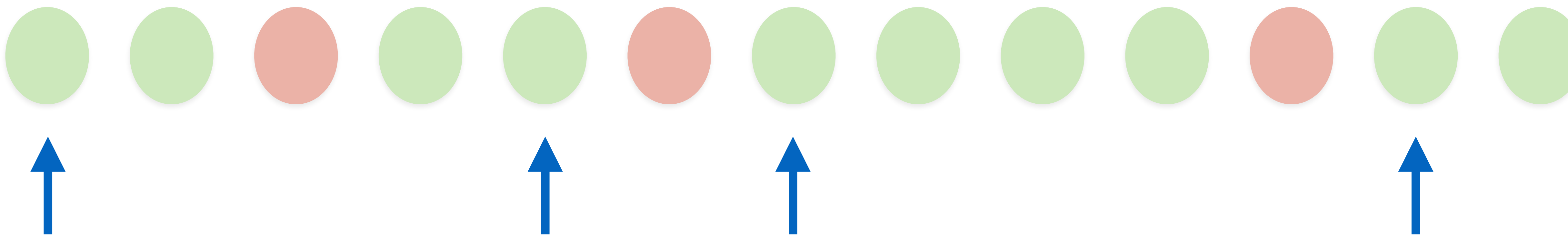
Suppose $T^* \not\subset G^*$



Case 2: T^* includes a bad circuit.

Success always!

Suppose $T^* \subset G^*$

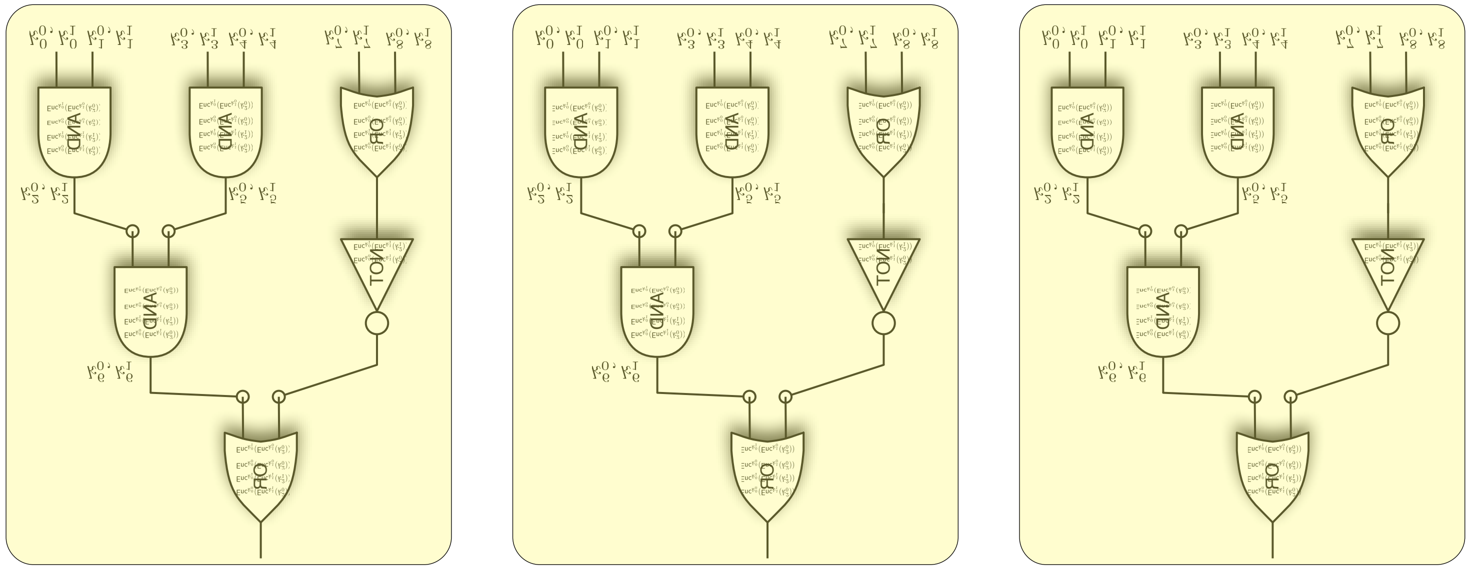


Case 3: T^* contains all good circuits.
Thus, evaluated circuits includes good+bad.

If outputs all the same, success (since ≥ 1 good)

If outputs differ, witness for check circuit.
success with pr $(1-2^{s})$

How to implement?

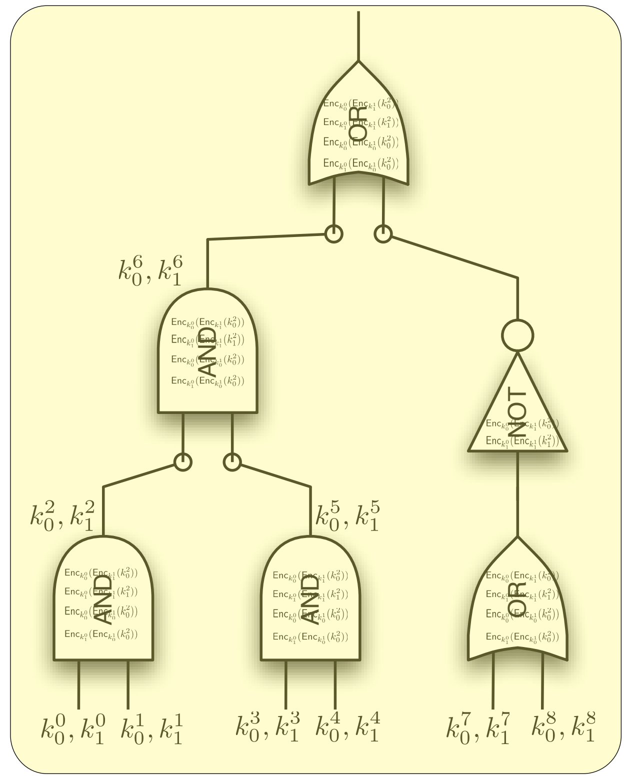


check(x,o$_1$,o$_k$)

x

$$b_i^0, b_i^1 \leftarrow \{0,1\}^k$$

Select random labels for output wire i



$$a_i^0, a_i^1 \leftarrow G$$

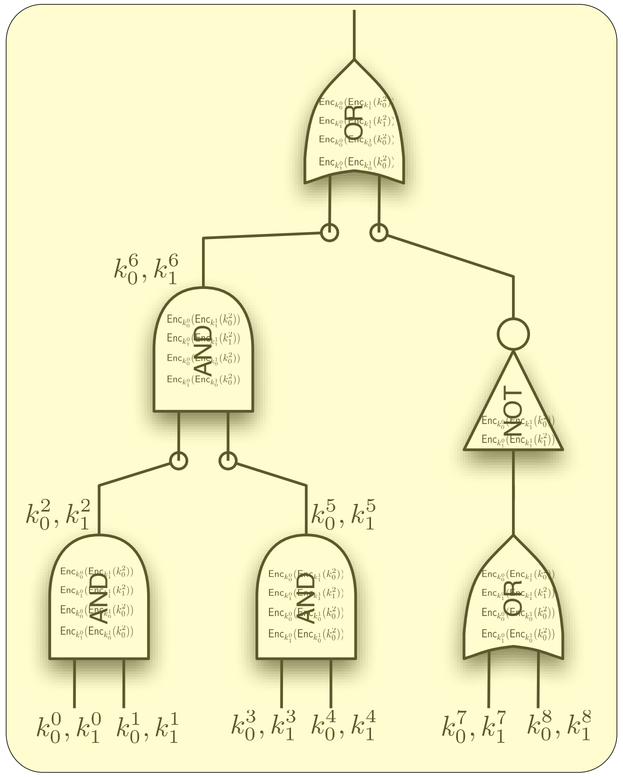
$$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$$

$$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$$

Select keys, make commitments to Alice's input wires.

1 Preparation

$$b_i^0, b_i^1 \leftarrow \{0,1\}^k$$



$$a_i^0, a_i^1 \leftarrow G$$

$$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$$

$$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$$

check(x,o_1,o_k)

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

The output wires b_i are hardcoded into the circuit by the Gen.

The inputs for x are shared with the main circuit. Use same a_i, different r_j.

If Bob can supply two witnesses for an output wire, Bob gets x.

Cut & choose used to test both main circuit and check circuits.

Must ensure that witness (o_1,o_k) does not come from cut-circuit.

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\left(\{w_{i,j}^b\}_{j,b}\right)$

Bob's 1 inp $\left(\{w_{i,j}^1\}_{j=1}^k, \chi_j\right)$

Bob's 0 inp $\{w_{i,j}^0\}_{j=1}^k, \chi_j$

y

Select challenge set J uniformly.

0 1 -,1 y_i, t_j

Oblivious* Transfer

for each input bit i

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

Run a cut-and-choose OT with keys for Bob's inputs. Special string Xj sent with OT that are not cut.

2 OT

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$$\left\{ (H(b_i^0), H(b_i^1)) \right\}_{i=1}^{m}$$

$$\left\{ (i, g^{a_i^0}, g^{a_i^1}) \right\}_{i=1}^{\ell}$$

$$\{(j, g^{r_j})\}_{j=1}^{k}$$

$$\{\mathsf{GC}_j\}_{j=1}^{k}$$

Send commitments to output table and Gen's input wire labels.

y

$\left(\{w_{i,j}^b\}_{j,b} \right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j \right)_{j \notin J}$

$\left\{ (H(b_i^0), H(b_i^1)) \right\}_{i=1}^{m}$

$\left\{ (i, g^{a_i^0}, g^{a_i^1}) \right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^{k}$

$\{\mathsf{GC}_j\}_{j=1}^{k}$

3 Commits

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

y

challenge set J

$\{\chi_j\}_{j \notin J}$

$$\left\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\right\}_{j \notin J, i}$$

Disclose challenge,
send Gen's input keys.

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\left\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\right\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1))\right\}_{i=1}^m$

$\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

4 Gen's Keys

x

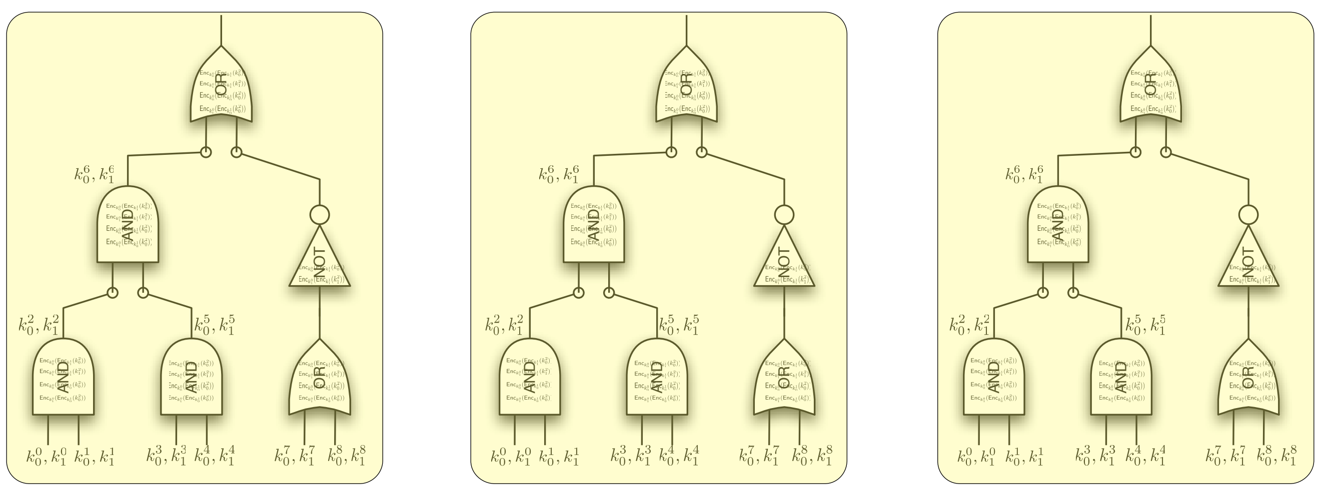
$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

y



$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\{(H(b_i^0), H(b_i^1))\}_{i=1}^m$

$\{(i, g^{a_i^0}, g^{a_i^1})\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

Evaluate all eval-circuits.
Record output wires.

5 Eval

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\hat{k}_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot \hat{r}_j})$

$\hat{k}_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot \hat{r}_j})$

check(x,o$_1$,o$_k$)

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

Use a maliciously secure protocol for check circuit. Gen reuses a$_i$ for inputs.

y

$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1)\right\}_{i=1}^m$

$\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

6 Cheat check

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\hat{k}_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot \hat{r}_j})$

$\hat{k}_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot \hat{r}_j})$

$$\xrightarrow{\{r_j\}_{j \in J}}$$

y

$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\left\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\right\}_{j \notin J, i}$

Use both keys to test GC$_j$ $\longrightarrow$ $\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1)\right\}_{i=1}^m$

Generate Eval's input keys $\longrightarrow$ $\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

Check these values $\longrightarrow$ $\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

7 Consistency of check circuits

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\hat{k}_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot \hat{r}_j})$

$\hat{k}_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot \hat{r}_j})$

$$\left[(g, g^{a_i^0}, g^{r_j}, k'_{i,j}) \in \mathsf{DH} \wedge (g, g^{a_i^0}, g^{\hat{r}_{\hat{j}}}, \hat{k}'_{i,\hat{j}}) \in \mathsf{DH}\right]_{j \notin J, \hat{j} \notin \hat{J}}$$

OR

$$\left[(g, g^{a_i^1}, g^{r_j}, k'_{i,j}) \in \mathsf{DH} \wedge (g, g^{a_i^1}, g^{\hat{r}_{\hat{j}}}, \hat{k}'_{i,\hat{j}}) \in \mathsf{DH}\right]_{j \notin J, \hat{j} \notin \hat{J}}$$

Run a SIGMA protocol to check Gen's input consistency.

y

$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\left\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\right\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1))\right\}_{i=1}^m$

$\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

8 Input consistency

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\hat{k}_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot \hat{r}_j})$

$\hat{k}_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot \hat{r}_j})$

y

$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\left\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\right\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1)\right\}_{i=1}^m$

$\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

Use either outputs + tables or the recovered x to send f(x,y) back.

9 Share output

Why is it secure?

x

$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

$\left(\{w_{i,j}^b\}_{j,b}\right)$

Bob's 1 inp $\left(\{w_{i,j}^1\}_{j=1}^k, \chi_j\right)$

Bob's 0 inp $\{w_{i,j}^0\}_{j=1}^k, \chi_j$

y

Select challenge set J uniformly.

0 1 -,1 y_i, t_j

Oblivious* Transfer

for each input bit i

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

When simulating for Bob*, run $\text{Simulator}_{\text{OT}}$ and get inputs + challenge

2 OT

x

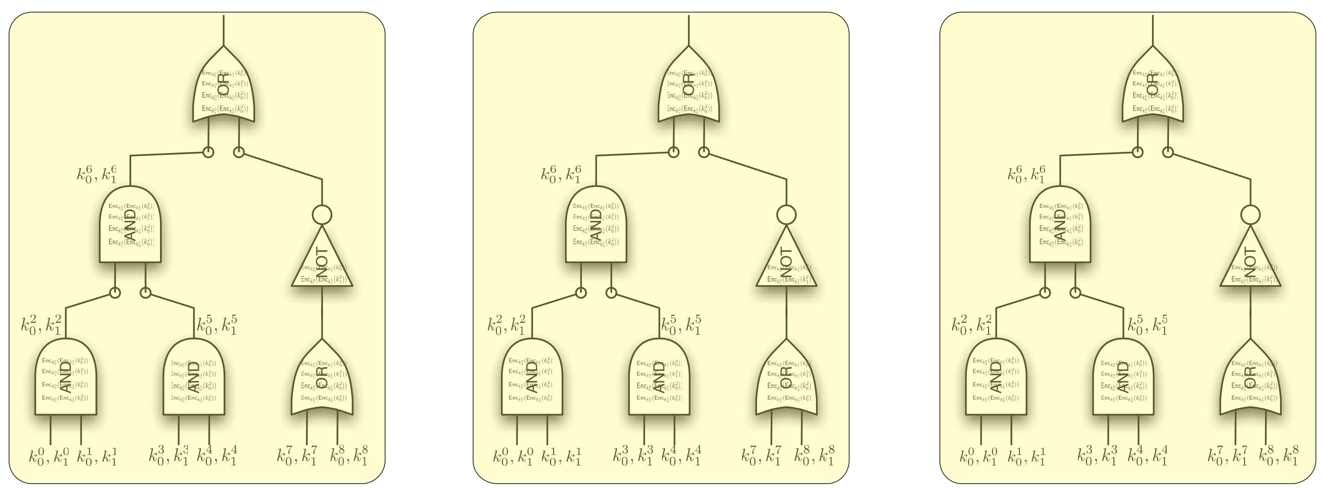
$a_i^0, a_i^1 \leftarrow G$

$b_i^0, b_i^1 \leftarrow \{0,1\}^k$

$k_{i,j}^0 \leftarrow H(g^{a_i^0 \cdot r_j})$

$k_{i,j}^1 \leftarrow H(g^{a_i^1 \cdot r_j})$

y



$\{o_{i,j}\}_j$

$\{k_{i,j} \leftarrow H(k'_{i,j})\}$

$\{k'_{i,j} \leftarrow g^{a_i^{x_i} \cdot r_j}\}_{j \notin J, i}$

$\left(\{w_{i,j}^b\}_{j,b}\right)_{j \in J}$

$\left(\{w_{i,j}^{y_i}\}_j, \chi_j\right)_{j \notin J}$

$\left\{(H(b_i^0), H(b_i^1)\right\}_{i=1}^m$

$\left\{(i, g^{a_i^0}, g^{a_i^1})\right\}_{i=1}^{\ell}$

$\{(j, g^{r_j})\}_{j=1}^k$

$\{\mathsf{GC}_j\}_{j=1}^k$

Can now program circuits.

5 Eval

Secure Garbling

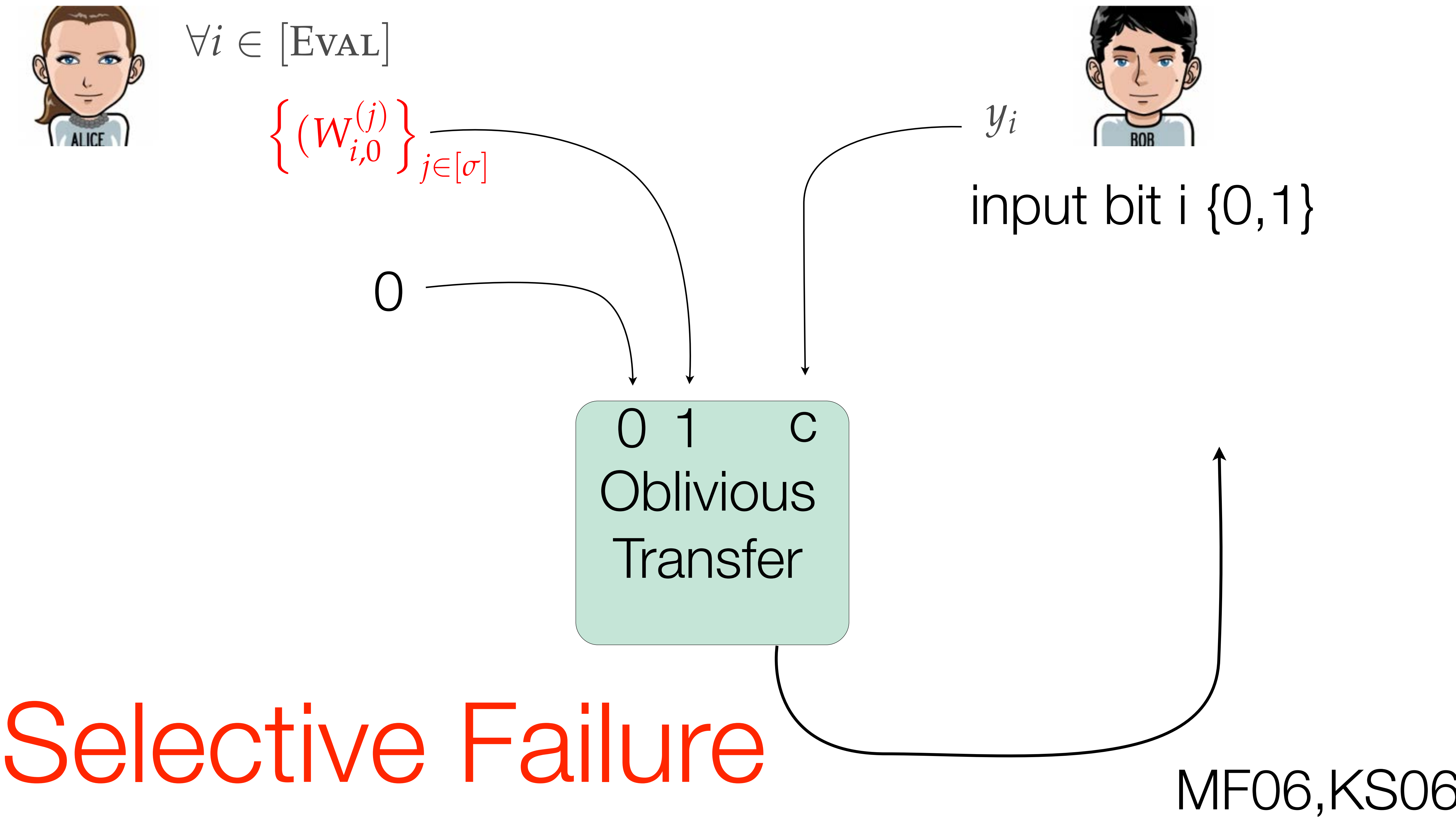
2. *Privacy: There exists a p.p.t. simulator algorithm S such that for all functions f and all inputs x, the following two distributions are computationally indistinguishable:*

$$\left\{(F, e, d) \leftarrow \textsc{gb}(1^k, f), X \leftarrow \textsc{en}(e, x) : (F, X, d)\right\}_{k \in \mathbb{N}} \approx_c \left\{S(1^k, f, f(x))\right\}_{k \in \mathbb{N}} \tag{2}$$

Protocol uses specific assumptions.

Open: Remove these (and have a faster protocol)

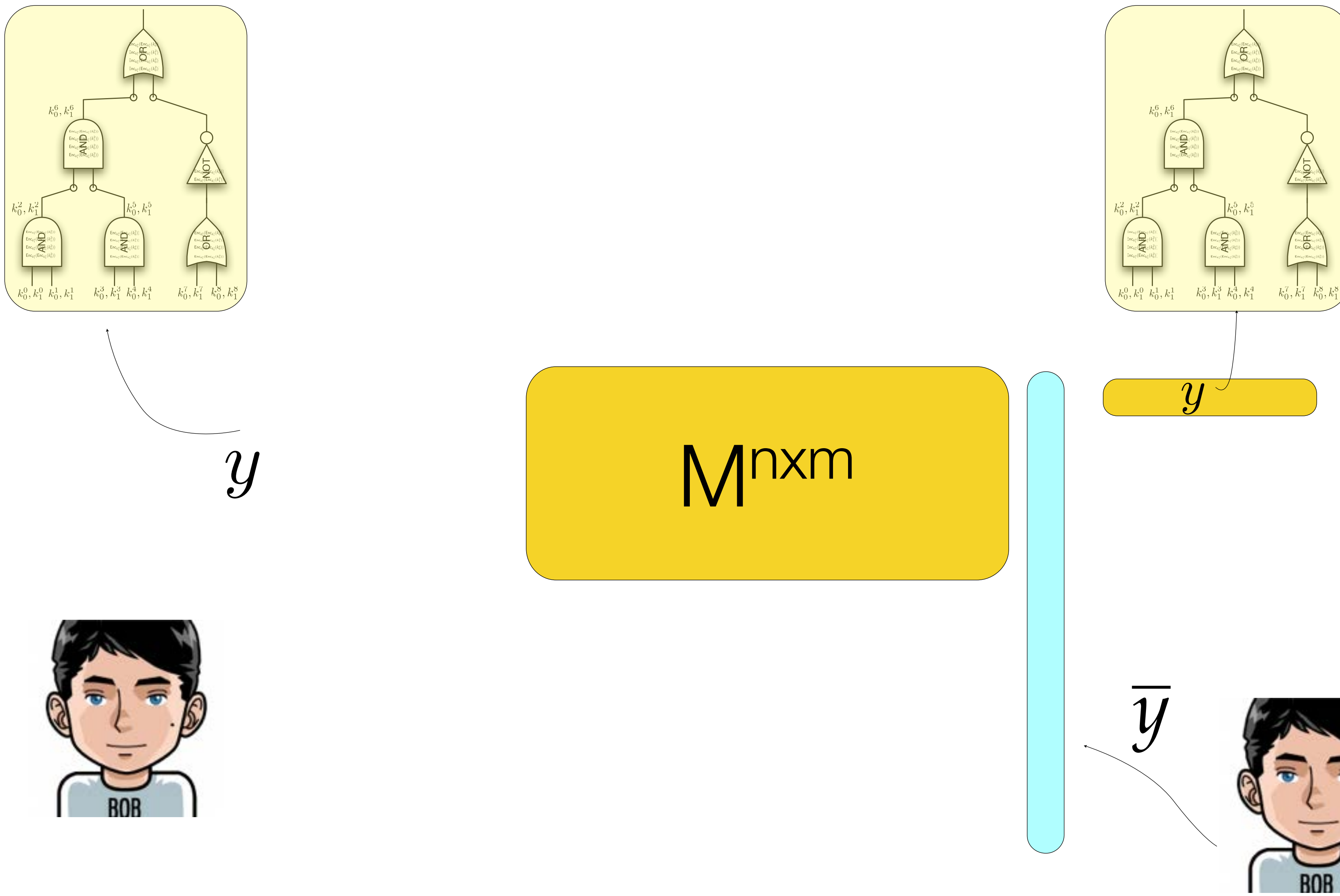
Selective Failure



$\forall i \in [\textsc{Eval}]$
$\left\{ (W_{i,0}^{(j)} \right\}_{j \in [\sigma]}$
0
y_i
input bit i {0,1}
0 1 c
Oblivious Transfer
Selective Failure
MF06,KS06

Encode Evaluator's input using error correcting code

[LP07]



y

$\max(4n, 8k)$

$M \cdot y$

Matrix M is k-probe resistant if $\mathsf{Ham}\left(\bigoplus_{i \in L} M_i\right) \geq k$
nonempty L

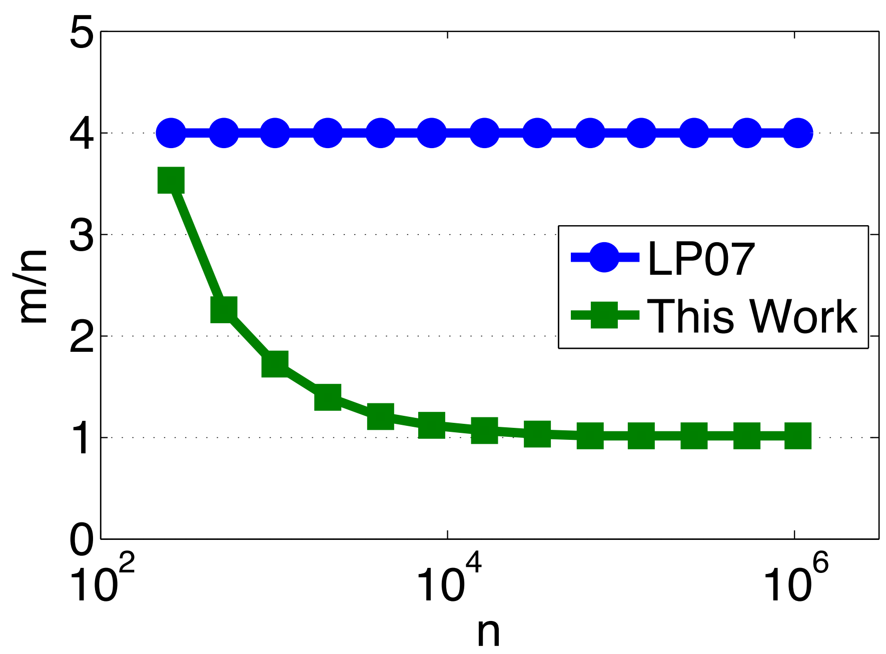
Pr[Eval aborts | y] - Pr[Eval aborts | y’] $\leq 2^{-k}$

One implementation

$$M \in \{0,1\}^{n \times m}$$

Explicit program to find M s.t.

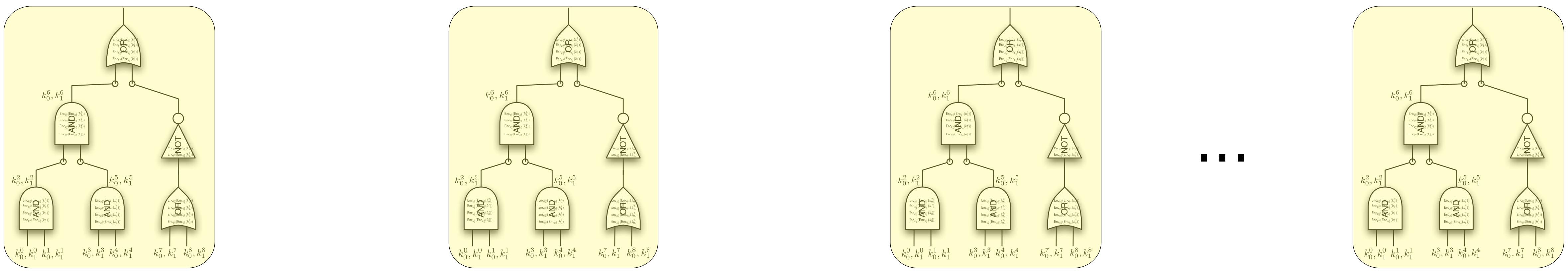
$$m \leq \lg(n) + n + k + k \cdot \max(\lg(4n), \lg(4k))$$



Input Consistency

Input consistency

copy: 1 2 3 k



x x x'

Did Alice use the same input
to each copy of the circuit?

	Input Consistency		2-Outputs		OT +
	sym	pke	sym	pke	
LP07 “blackbox”	$\Theta(k^2 n)$		$\Theta(k^2 n)$		OWF
K08	$\Theta(k^2 n)$		$\Theta(kn)$	$\Theta(k)$	DLOG
LP11	$\Theta(kn)$	$\Theta(kn)$			DLOG
SS11,	$\Theta(kn)$	$\Theta(kn)$	$\Theta(kn)$	$\Theta(kn)$	DLOG
KSS12	“	“	“	$\Theta(k)$	DLOG
SS13	$\Theta(kn)$		$\Theta(kn)$		OWF

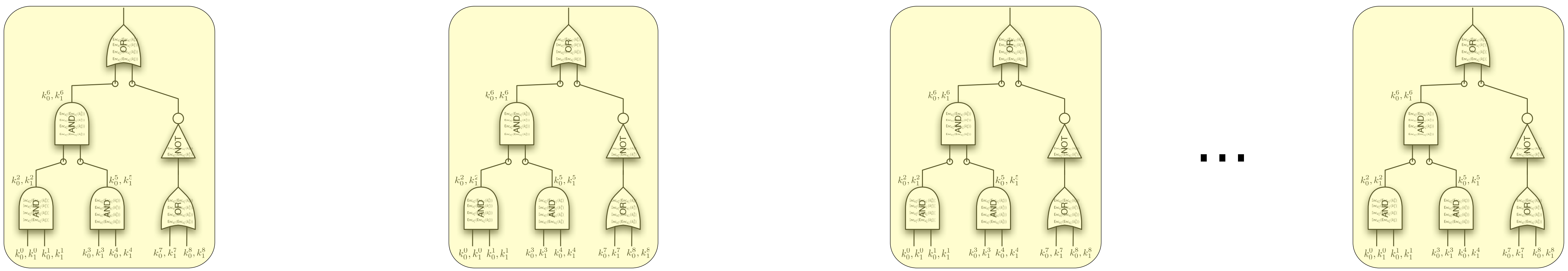
Result:

Fewer Assumptions

Faster Protocol

Input consistency

copy: 1 2 3 ... k



x x x

"WI proof that input $x^{(1)} = \cdots = x^{(k)}$ "

[LP10, SS11]
Sigma protocols

Inspiration

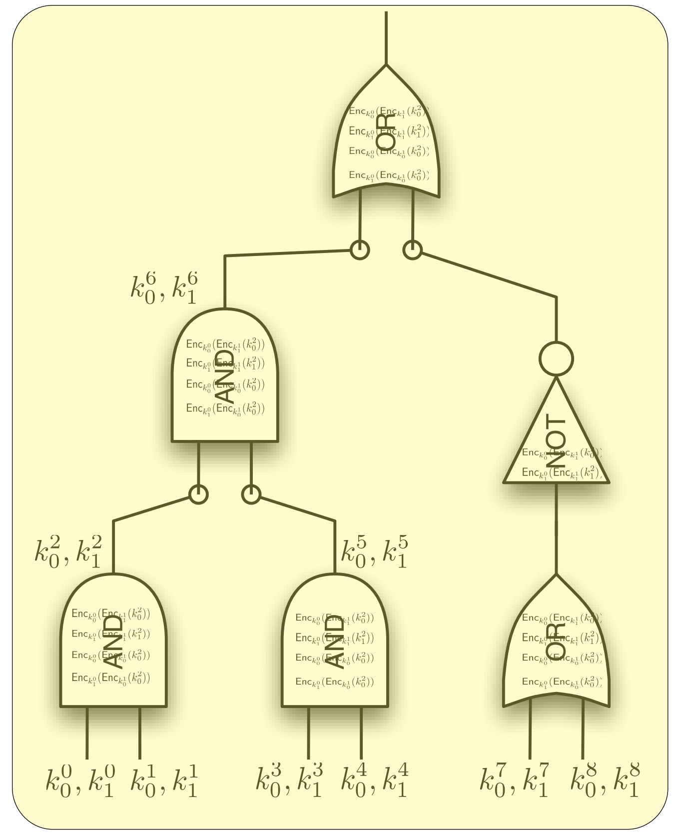
"WI proof that input $x^{(1)} = \cdots = x^{(k)}$ "

Are there better algorithms to implement this proof?

Recursion?

Our approach:

input
consistency
circuit g(x)



x

y

Inspiration

"WI proof that input $x^{(1)} = \cdots = x^{(k)}$ "

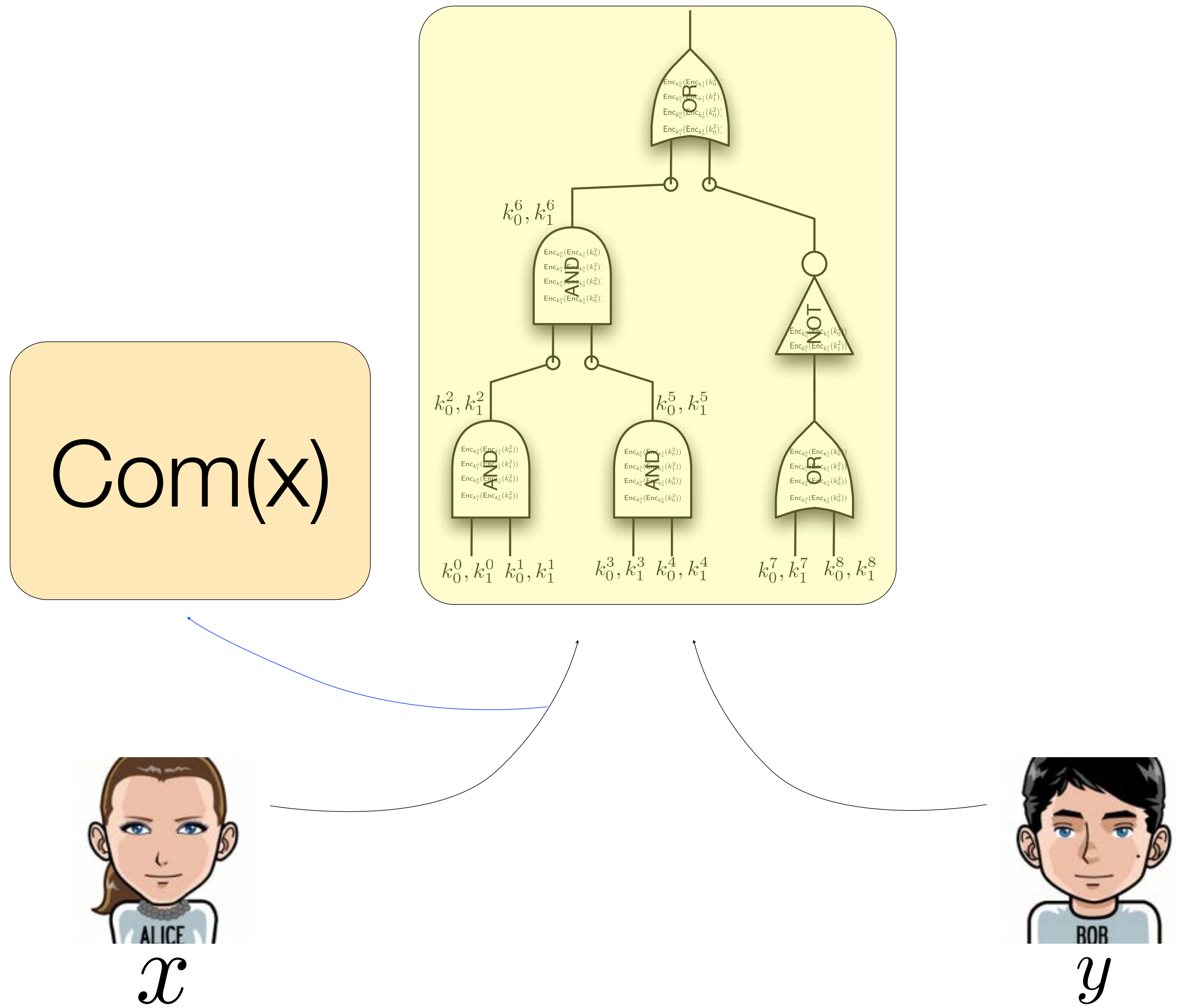
Proof: $g(x^{(1)}), \ldots, g(x^{(k)})$

For what choices of g will this proof be sound + WI?

g should be hiding

g should be "binding"

Should be: hiding “binding”

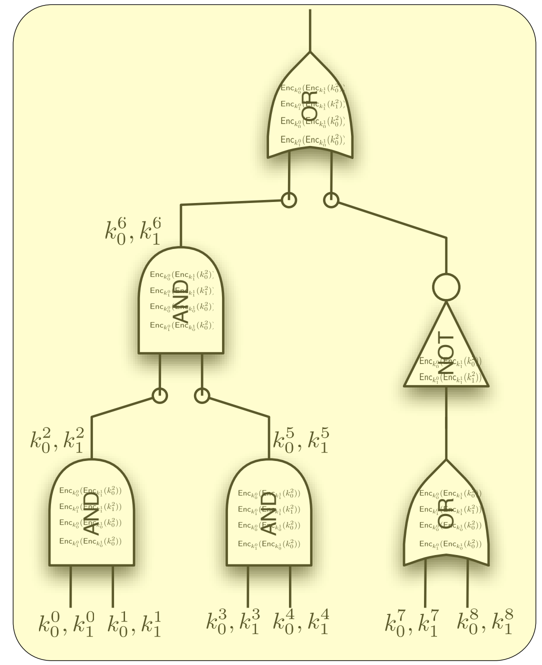


Obvious candidate: Commitment scheme

Problem: could be a large circuit

Should be: hiding "binding"

$h_M(x)$



x

y

Next candidate: 2-Universal hash function

2-universal hash function

$$\boxed{\quad x|r \quad} \left[M \right] = \boxed{y}$$

Hiding: By left-over hash lemma, (x|r)M will be hiding for large enough r.

2-universal hash function

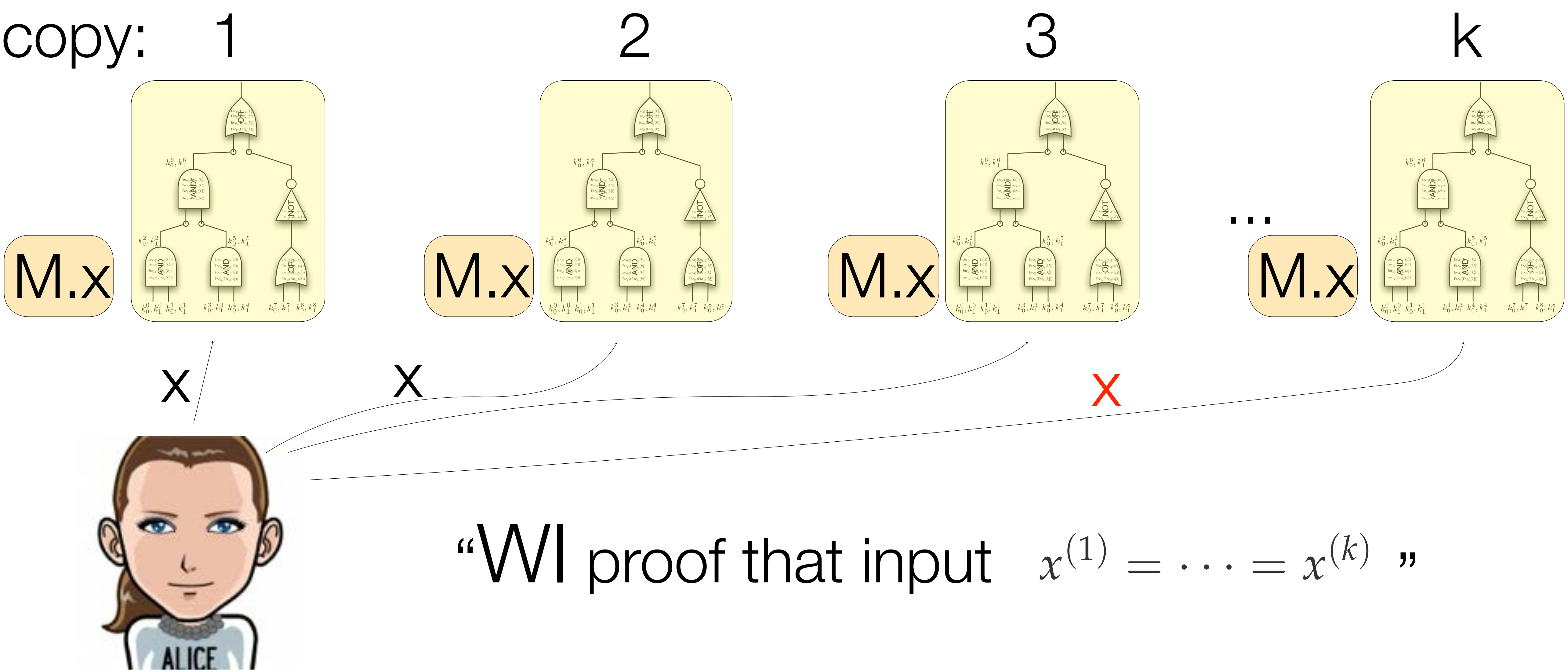
$$\boxed{x|r}\,\left[M\right] = \boxed{y}$$

Binding: For any $x \neq x'$ pr over random choice of M :

$$\Pr[M(x) = M(x')] < B^{-1}$$

Idea: Pick the function M after GEN has committed to inputs.

New input consistency



Cost of each evaluation M.x

M.x $\qquad n^2$ AND gates

Why restrict ourselves to Garbled circuits for M.x ?

Note that h_M is homomorphic

$$h_M(\pi) + h_M(x) = h_M(x + \pi)$$

$x^{(1)}, \ldots, x^{(k)} \in \{0,1\}^m$

$\pi^{(1)}, \ldots, \pi^{(k)}$

$\mathsf{com}(x^{(1)}), \ldots, \mathsf{com}(x^{(k)})$

$\longrightarrow$

$\mathsf{com}(x^{(1)} + \pi^{(1)}), \ldots, \mathsf{com}(x^{(k)} + \pi^{(k)})$

Pick h_M

$\longleftrightarrow$

$h_M(\pi^{(1)}), \ldots, h_M(\pi^{(k)})$

$\longrightarrow$

challenge $s^{(1)}, \ldots, s^{(k)}$

$\longleftarrow$

decommit $\mathsf{com}(\pi^{(j)})$ if $s^{(j)} = 0$

decommit $\mathsf{com}(x^{(j)} + \pi^{(j)})$ else

$\longrightarrow$

How to implement?

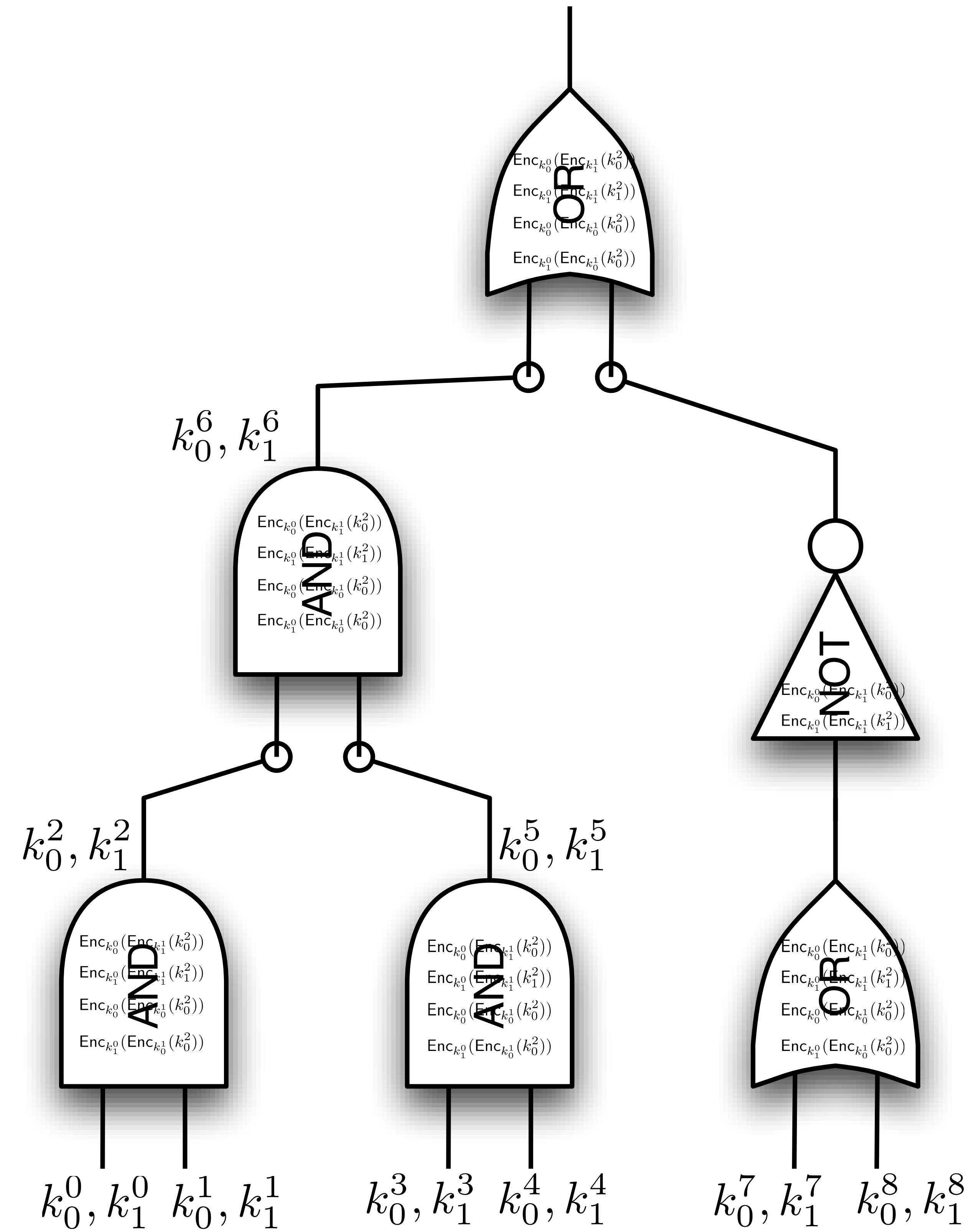
1. Setup
2. Commit to Input Labels
3. Pick H,M
4. Eval Input OT
5. Circuit OT
6. Garbling-Evaluation
7. Input Consistency

garbled circuit:

each wire has a key pair

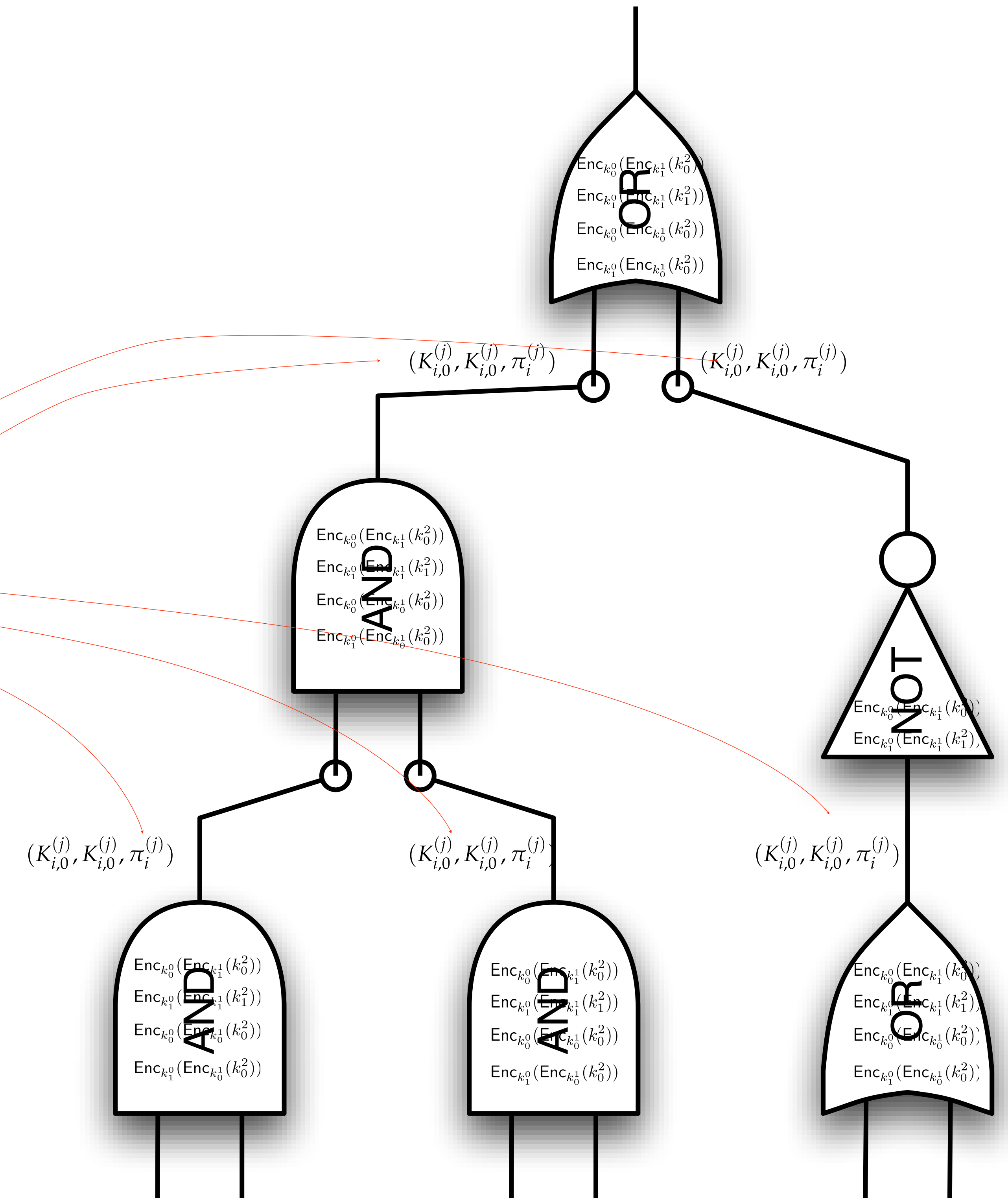
each gate has a table

$$\rho$$



Use PRF and seed to generate all wires for circuit j

$\rho^{(j)}$



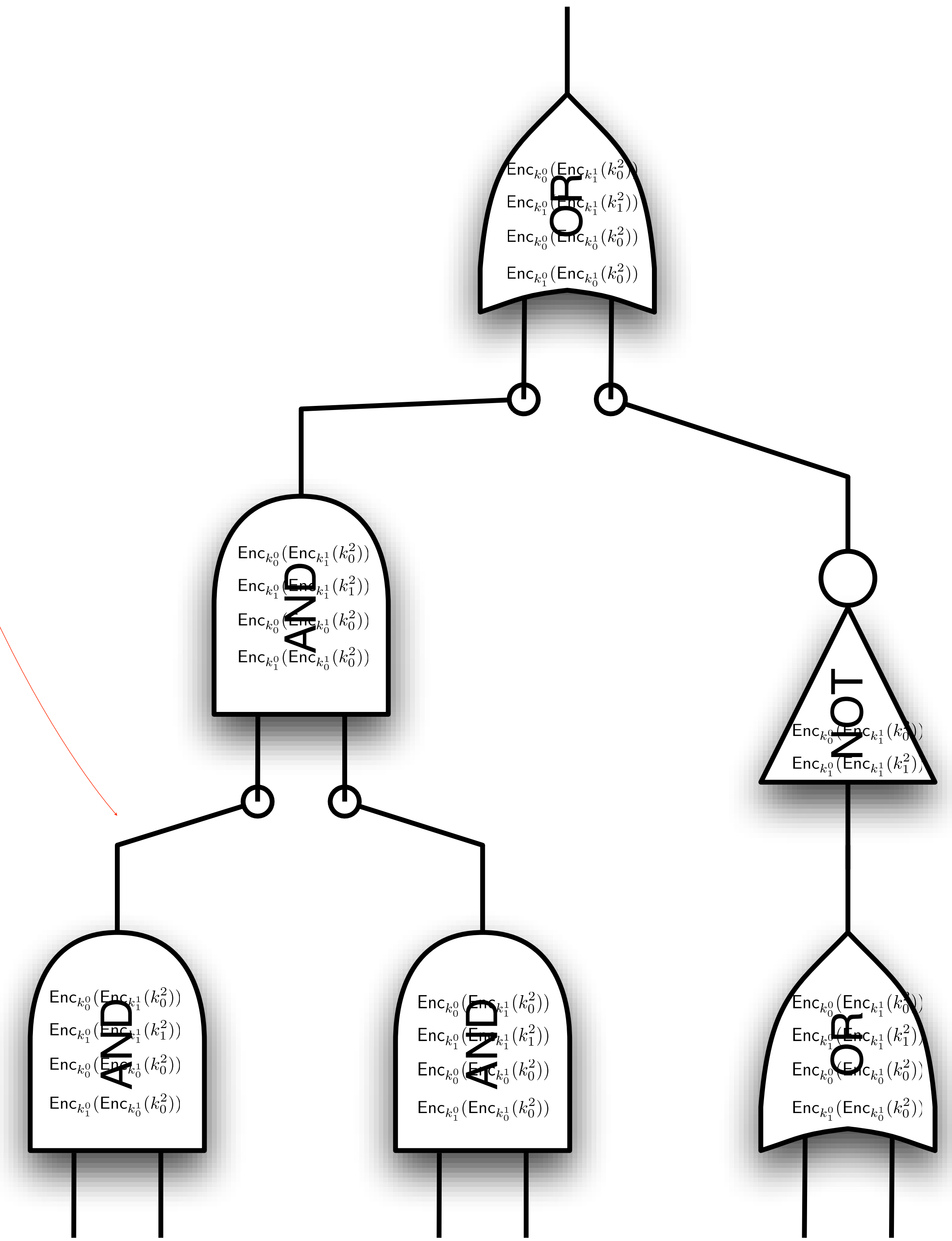
j^th copy

$$(K_{i,0}^{(j)}, K_{i,0}^{(j)}, \pi_i^{(j)})$$

labels for wire i

$$W_{i,b}^{(j)} = (K_{i,b}^{(j)}, b + \pi_i^{(j)})$$

key locator for wire i



x

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

y

$\overline{y} \leftarrow My$

1 Setup

x

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

y

$\overline{y} \leftarrow My$

$$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi_i^{(j)}};\theta_i^{(j)}), \mathsf{com}(W^{(j)}_{i,1\oplus\pi_i^{(j)}};\theta_i^{(j)})\}_{i\in[\textsc{Gen}]}$$

gen inputs

$$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W^{(j)}_{m_1+i,1})\}_{i\in[\textsc{Eval}]}$$

eval inputs

$\rho^{(j)}$
used to commit

$$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\bar{x}_i};\gamma_i^{(j)})\}_{i\in[\textsc{Gen}]}.$$

independent randomness here!

2 Commit keys

x

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

M

y

$\overline{y} \leftarrow My$

$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi_i^{(j)}}; \theta_i^{(j}$

$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W$

$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\bar{x}_i}; \gamma_i^{(j)})\}_{i\in[\mathrm{GE}}$

pick H

3 Flip

x

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

H, M

$\forall i \in [\textsc{Eval}]$

$\left\{(W_{i,0}^{(j)}\right\}_{j\in[\sigma]}$

$\left\{(W_{i,1}^{(j)}\right\}_{j\in[\sigma]}$

$\overline{y}_i$

0 1 c

Oblivious Transfer

y

$\overline{y} \leftarrow My$

$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi_i^{(j)}}; \theta_i^{(j)})$

$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W$

$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\bar{x}_i}; \gamma_i^{(j)})\}_{i\in[\textsc{Gen}]}$

H, M

$\left\{W^{(j)}_{i,\overline{y}_i}\right\}_{i\in\mathrm{Eval}}$

4 Input key OT

x

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

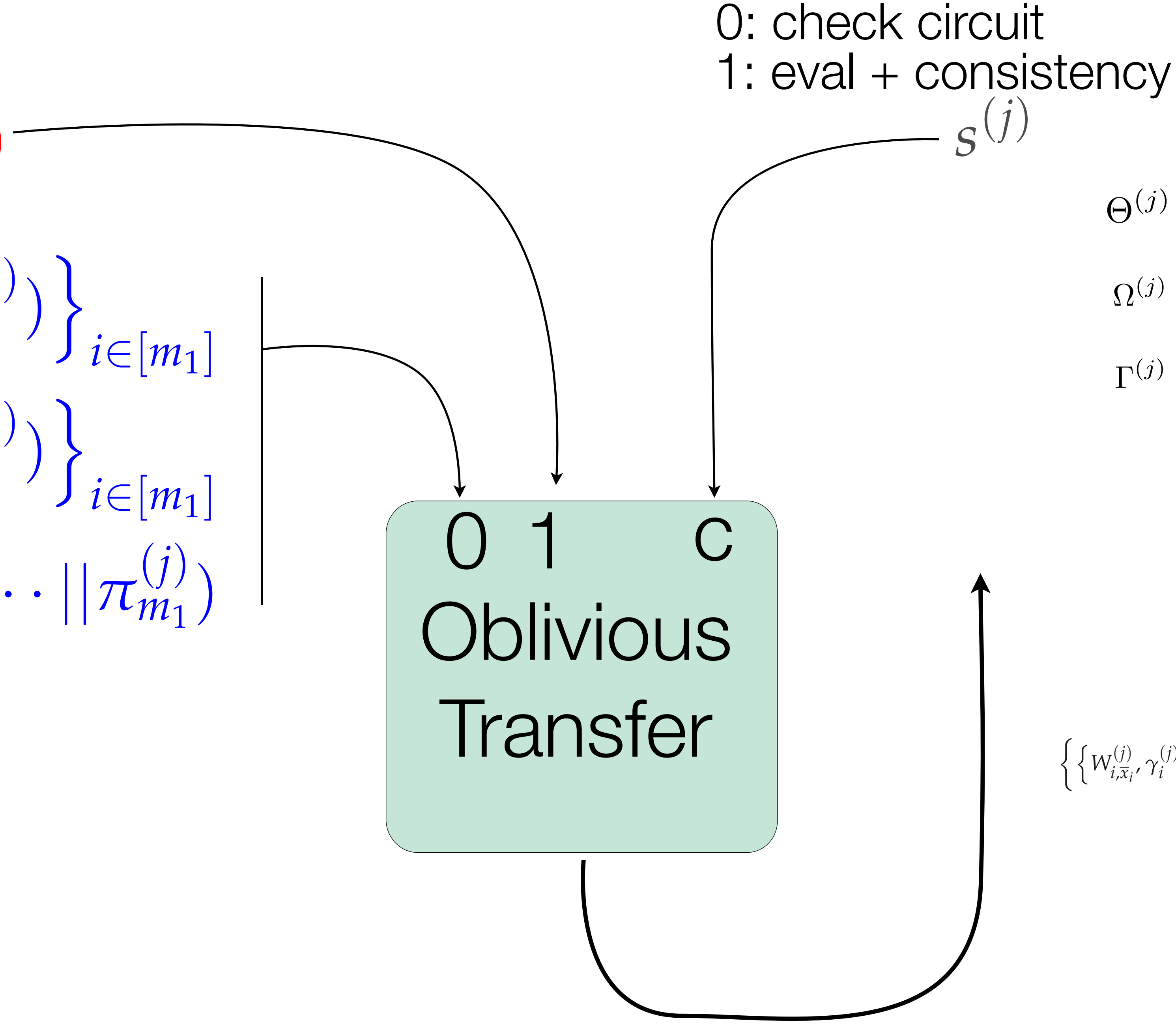
pick $\rho^{(j)}$

H, M

$\left\{ (W^{(j)}_{i,\bar{x}_i}, \gamma^{(j)}_i) \right\}_{i \in [m_1]}$

$\left\{ (W^{(j)}_{i,\bar{x}_i}, \theta^{(j)}_i) \right\}_{i \in [m_1]}$

$h^{(j)}_{\pi} = H \cdot (\pi^{(j)}_1 || \pi^{(j)}_2 || \cdots || \pi^{(j)}_{m_1})$



y

$\overline{y} \leftarrow My$

$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi^{(j)}_i}; \theta^{(j)}_i)$

$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W$

$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\bar{x}_i}; \gamma^{(j)}_i)\}_{i\in[\textsc{Gen}]}$

H, M

$\left\{ W^{(j)}_{i,\bar{y}_i} \right\}_{i \in \mathrm{Eval}}$

$\left\{ \rho^{(j)} \right\}_{s^{(j)}=1}$

$\left\{ \left\{ W^{(j)}_{i,\bar{x}_i}, \gamma^{(j)}_i, W^{(j)}_{i,\bar{x}_i}, \theta^{(j)}_i \right\}_{i\in[m_1]}, H(\pi^(j)_1||\cdots) \right\}_{s^{(j)}=}$

5 Circuit OT

x

$\overline{x} \leftarrow x || e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

H, M

y

$\overline{y} \leftarrow My$

$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi_i^{(j)}}; \theta_i^{(j)})$

$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W_{m}$

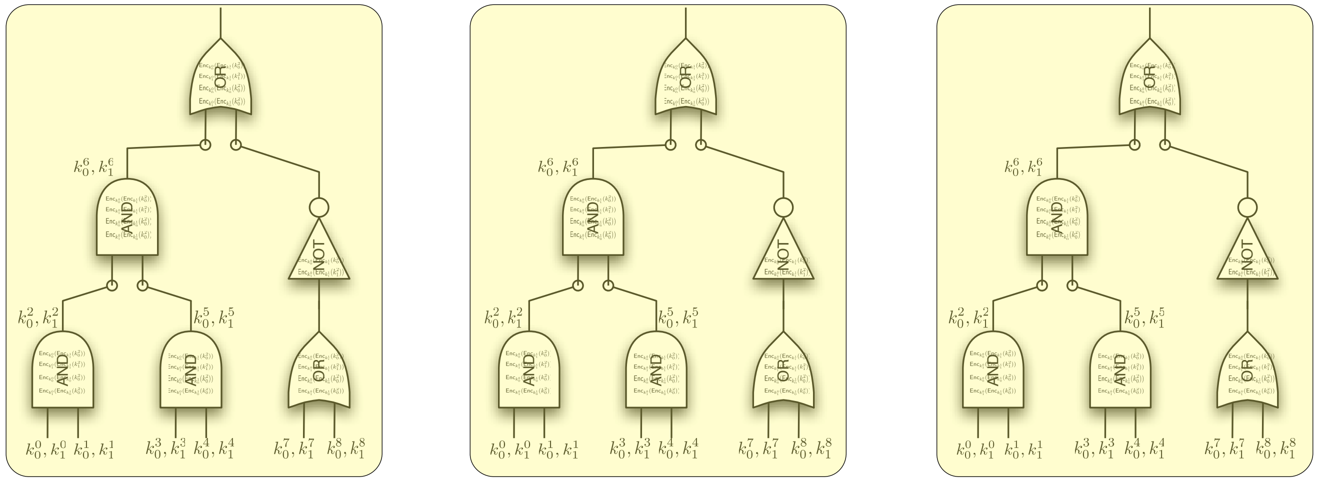
$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\overline{x}_i}; \gamma_i^{(j)})\}_{i\in[\textsc{Gen}]}$

H, M

$\left\{W^{(j)}_{i,\overline{y}_i}\right\}_{i\in\mathrm{Eval}}$

$\left\{\rho^{(j)}\right\}_{s(j)=1}$

$\left\{\left\{W^{(j)}_{i,\overline{x}_i}, \gamma_i^{(j)}, W^{(j)}_{i,\overline{x}_i}, \theta_i^{(j)}\right\}_{i\in[m_1]}, H(\pi^{(j)})_1 || \cdots\right\}_{s(j)=}$



Eval either checks circuits using { p(j) }sj=1

OR evals circuits using wire labels.

$\left\{GC^{(j)}, \Theta^{(j)}, \Omega^{(j)}\right\}$

$h_\pi^{(j)}$

Eval can abort on fail.

6 Garbling

$$x$$

$\overline{x} \leftarrow x || e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

H, M

$$y$$

$\overline{y} \leftarrow My$

For all check (j), let $W_i^{(j)} = (K_i^{(j)}, \delta_i^{(j)})$

Compute

$$h_{\bar{x}}^{(j)} = h_{\pi}^{(j)} \oplus H \cdot (\delta_1^{(j)} || \delta_2^{(j)} || \cdots || \delta_{m_1}^{(j)})$$

For all a,b in check, verify:

$$h_{\bar{x}}^{(a)} = h_{\bar{x}}^{(b)}$$

Eval can abort on fail.

$\Theta^{(j)} = \{\mathsf{com}(W_{i,0\oplus\pi_i^{(j)}}^{(j)}; \theta_i^{(j)})$

$\Omega^{(j)} = \{\mathsf{com}(W_{m_1+i,0}^{(j)}), \mathsf{com}(W_{m}$

$\Gamma^{(j)} = \{\mathsf{com}(W_{i,\bar{x}_i}^{(j)}; \gamma_i^{(j)})\}_{i\in[\textsc{Gen}]}$

H, M

$\left\{ W_{i,\overline{y}_i}^{(j)} \right\}_{i \in \mathrm{Eval}}$

$\left\{ \rho^{(j)} \right\}_{s^{(j)}=1}$

$\left\{ \left\{ W_{i,\overline{x}_i}^{(j)}, \gamma_i^{(j)}, W_{i,\overline{x}_i}^{(j)}, \theta_i^{(j)} \right\}_{i\in[m_1]}, H(\pi^(j)_1 || \cdots) \right\}_{s^{(j)}=}$

7 Input consistency

$$x$$

$\overline{x} \leftarrow x||e$

$r \leftarrow \{0,1\}^{2k+\lg(k)}$

pick $\rho^{(j)}$

H, M

$$y$$

$\overline{y} \leftarrow My$

Send majority of output if all checks pass.

$\Theta^{(j)} = \{\mathsf{com}(W^{(j)}_{i,0\oplus\pi_i^{(j)}}; \theta_i^{(j)})$

$\Omega^{(j)} = \{\mathsf{com}(W^{(j)}_{m_1+i,0}), \mathsf{com}(W^{(j)}_{m}$

$\Gamma^{(j)} = \{\mathsf{com}(W^{(j)}_{i,\bar{x}_i}; \gamma_i^{(j)})\}_{i\in[\text{Gen}]}$

H, M

$\left\{W^{(j)}_{i,\overline{y}_i}\right\}_{i\in\text{Eval}}$

$\left\{\rho^{(j)}\right\}_{s^{(j)}=1}$

$\left\{\left\{W^{(j)}_{i,\overline{x}_i}, \gamma_i^{(j)}, W^{(j)}_{i,\overline{x}_i}, \theta_i^{(j)}\right\}_{i\in[m_1]}, H(\pi^(j)_1||\cdots)\right\}_{s^{(j)}=}$

8 Majority

Performance

Evaluations

[KSS12]

circuit	gates	(non-XOR)	time (sec)	comm.
EDT-4095	5.9B	(2.4B)	9,042	18 TB
RSA-256	0.93B	(0.33B)	1,437	3 TB
1024-AES128	32M	(9.3M)	49	74 GB

roughly 650k gates/second total thruput

1.7m g/sec garble rate

60% of time spent on network

Texas Advanced Computing Center. 32 nodes; each node: 2 Xeon E5-2680 2.7Ghz (each has 8 cores), 32GB

AES 2^{-80} security

[KSS12]

		Gen (sec)	Eval (sec)	Comm (KB)
OT	comp	45.8±0.09%	34.0±0.2%	5,516
	comm	0.1± 1%	11.9±0.6%	
Gen.	comp	35.6± 0.5%	–	3
	comm	–	35.6±0.5%	
Inp. Chk	comp	–	1.75±0.2%	266
	comm	–	–	
Evl.	comp	14.9± 0.6%	32.4±0.4%	28,781
	comm	18.2± 1%	3.2±0.8%	
Total	comp	96.3± 0.3%	68.0±0.2%	44,805
	comm	18.3± 1%	50.8±0.4%	

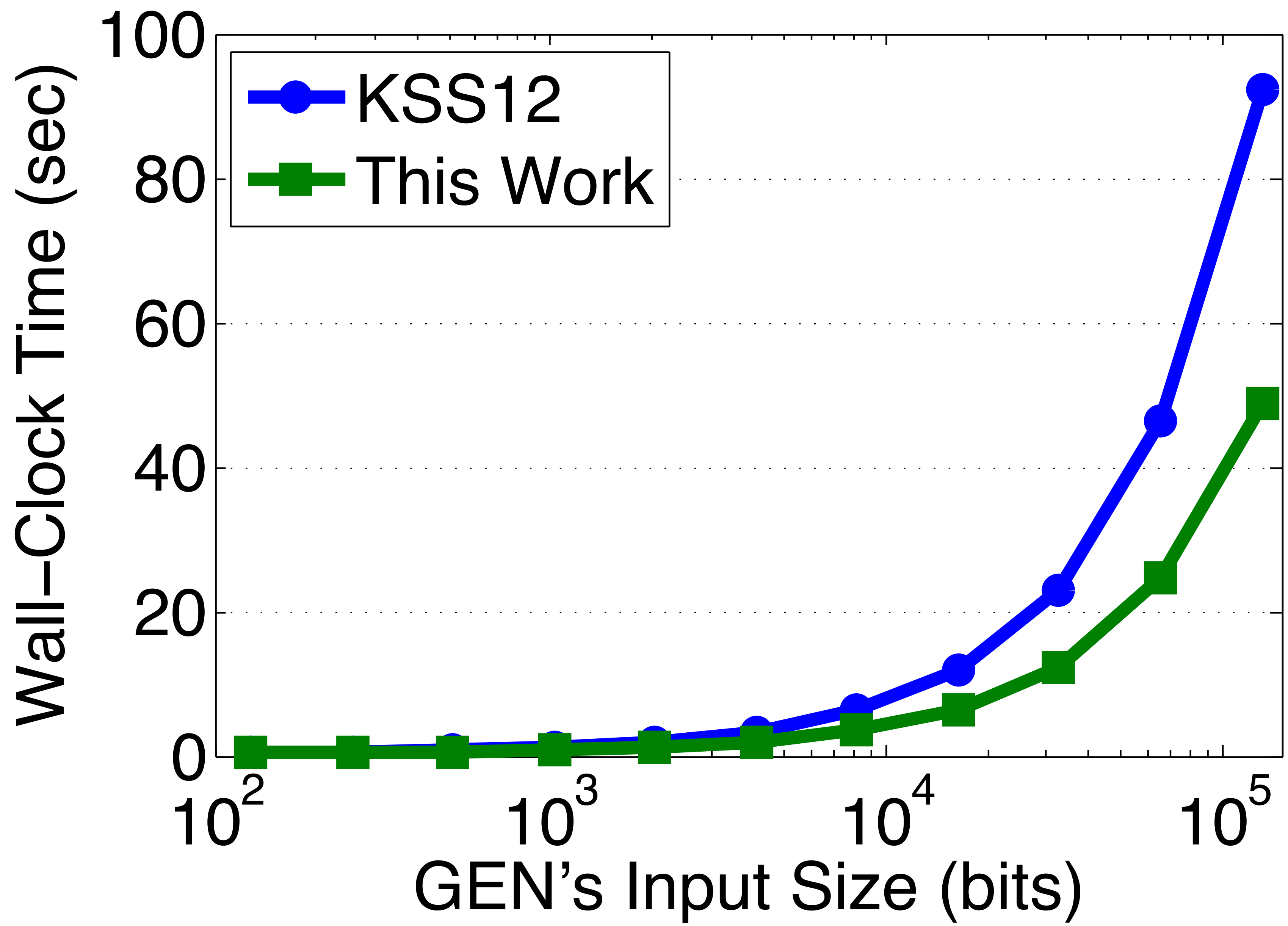
1 core

Parallel Impl

node #	4		16		64		256	
	Gen	Evl	Gen	Evl	Gen	Evl	Gen	Evl
OT	12.56±0.1%	8.41±0.1%	4.06±0.1%	2.13±0.2%	1.96±0.1%	0.58±0.2%	0.64±0.1%	0.19±0.2%
Gen.	8.18±0.4%	–	1.92±0.7%	–	0.49±0.4%	–	0.14± 1%	–
Inp. Chk	–	0.42± 4%	–	0.10± 10%	–	–	–	–
Evl.	3.3± 4%	7.08± 1%	0.80± 10%	1.58± 4%	0.23± 17%	0.37± 7%	0.12±0.5%	0.05±0.6%
Inter-com	4± 5%	13.2±0.3%	0.93± 10%	4.08±0.8%	0.31± 20%	1.98± 1%	0.11± 40%	0.72±0.2%
Intra-com	0.17± 30%	0.23± 20%	0.18± 8%	0.25± 6%	0.45± 20%	0.48± 15%	0.34± 30%	0.34± 30%
Total time	28.3±0.3%	29.4±0.3%	7.90±0.5%	8.17±0.4%	3.45± 2%	3.44± 2%	1.4± 10%	1.3± 9%

HEKM11: 1.6s

honest-but-curious



Parameterized AES function, $f(x, (y_1,\ldots,y_n) = AES_x(y_1),\ldots,AES_x(y_n)$

AESNI

```
void AES_128_Key_Expansion (const unsigned char *userkey,
                            unsigned char *key)
{
    __m128i temp1, temp2;
    __m128i *Key_Schedule = (__m128i*)key;
    temp1 = _mm_loadu_si128((__m128i*)userkey);
    Key_Schedule[0] = temp1;

    // __builtin_ia32_aeskeygenassist128((temp1), (0x1));

    temp2 = _mm_aeskeygenassist_si128 (temp1 ,0x1);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[1] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x2);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[2] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x4);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[3] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x8);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[4] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x10);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[5] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x20);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[6] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x40);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[7] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x80);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[8] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x1b);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[9] = temp1;
    temp2 = _mm_aeskeygenassist_si128 (temp1,0x36);
    temp1 = AES_128_ASSIST(temp1, temp2);
    Key_Schedule[10] = temp1;


}
```

```
// in: pointer to 16 bytes
// out: pointer to 16 bytes
// key: full 10-round keyschedule
void AES_prf(const unsigned char *in, unsigned char *out,
const unsigned char *key)
{
    __m128i tmp;
    tmp = _mm_loadu_si128 (&((__m128i*)in)[0]);
    tmp = _mm_xor_si128 (tmp,((__m128i*)key)[0]);

    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[1]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[2]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[3]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[4]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[5]);

    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[6]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[7]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[8]);
    tmp = _mm_aesenc_si128 (tmp,((__m128i*)key)[9]);

    tmp = _mm_aesenclast_si128 (tmp,((__m128i*)key)[10]);
    _mm_storeu_si128 (&((__m128i*)out)[0],tmp);

}
```

```
abhis-MacBook-Pro:aes abhi$ ./aesni
cpu support: 2000000
test clicks: 36
clicks: 2213795401 221.379540
```

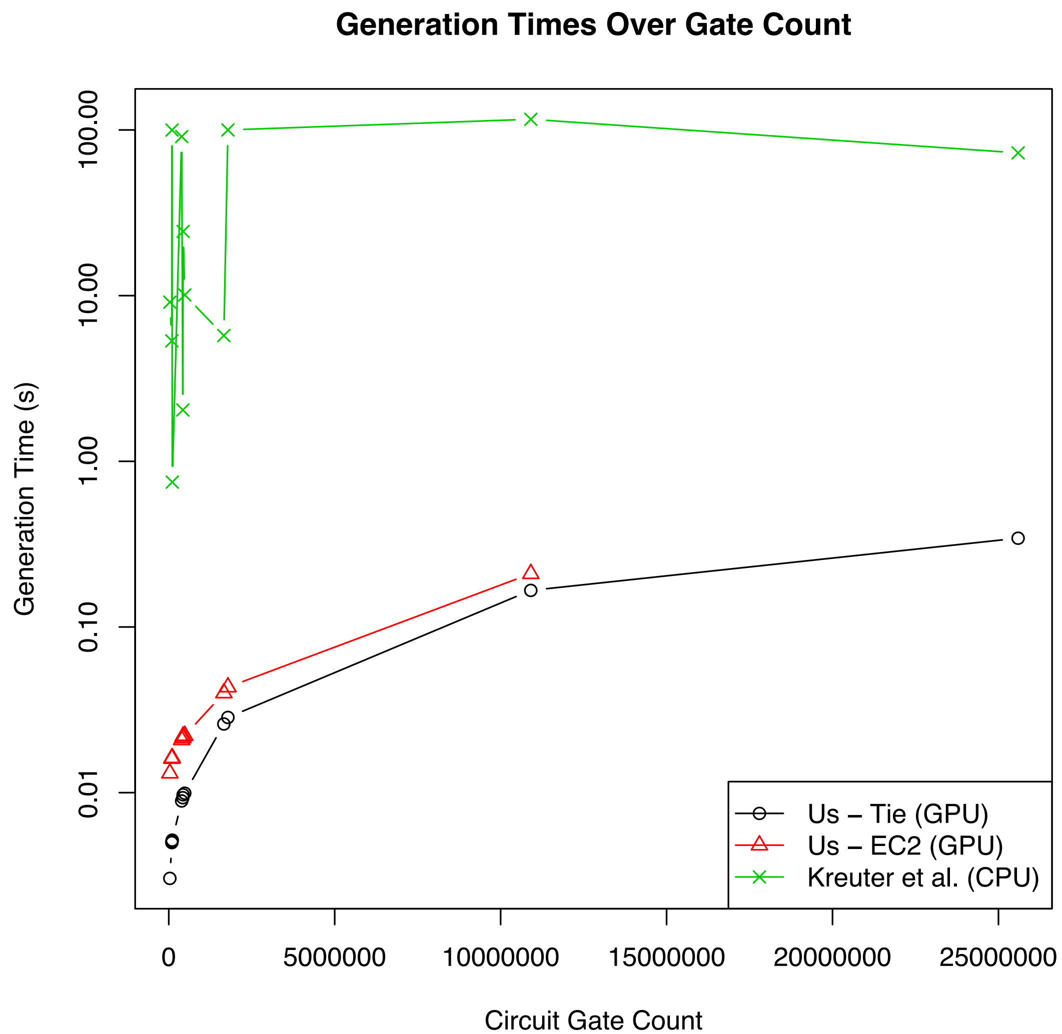
AESNI

```
void KDF128(const uint8_t *in, uint8_t *out, const uint8_t *key)
{
    ALIGN16 uint8_t KEY[16*11];
    ALIGN16 uint8_t PLAINTEXT[64];
    ALIGN16 uint8_t CIPHERTEXT[64];

    AES_128_Key_Expansion(key, KEY);
    _mm_storeu_si128(&((__m128i*)PLAINTEXT)[0],*(__m128i*)in);
    AES_ECB_encrypt(PLAINTEXT, CIPHERTEXT, 64, KEY, 10);
    _mm_storeu_si128((__m128i*)out,((__m128i*)CIPHERTEXT)[0]);
}

void KDF256(const uint8_t *in, uint8_t *out, const uint8_t *key)
{
    ALIGN16 uint8_t KEY[16*15];
    ALIGN16 uint8_t PLAINTEXT[64];
    ALIGN16 uint8_t CIPHERTEXT[64];

    AES_256_Key_Expansion(key, KEY);
    _mm_storeu_si128(&((__m128i*)PLAINTEXT)[0],*(__m128i*)in);
    AES_ECB_encrypt(PLAINTEXT, CIPHERTEXT, 64, KEY, 14);
    _mm_storeu_si128((__m128i*)out,((__m128i*)CIPHERTEXT)[0]);
}
```

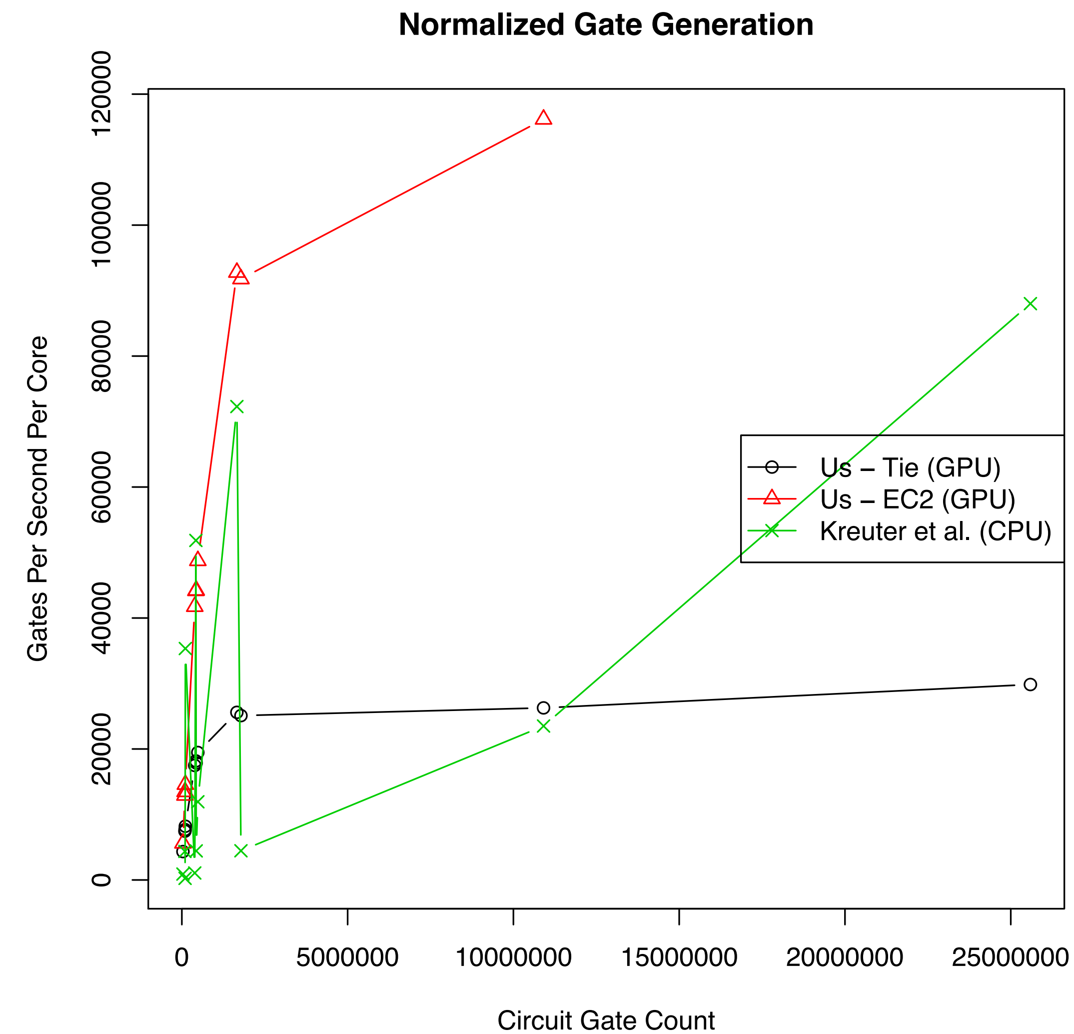
AES-NI

	size (gate)	AES-NI (sec)	SHA-256 (sec)	Ratio (%)
AES	49,912	0.12± 1%	0.15± 1%	78.04
Dot_4^{64}	460,018	1.11±0.4%	1.41±0.5%	78.58
RSA-32	1,750,787	4.53±0.5%	5.9±0.8%	76.78
EDT-255	15,540,196	42.0±0.5%	57.6± 1%	72.92

GPU

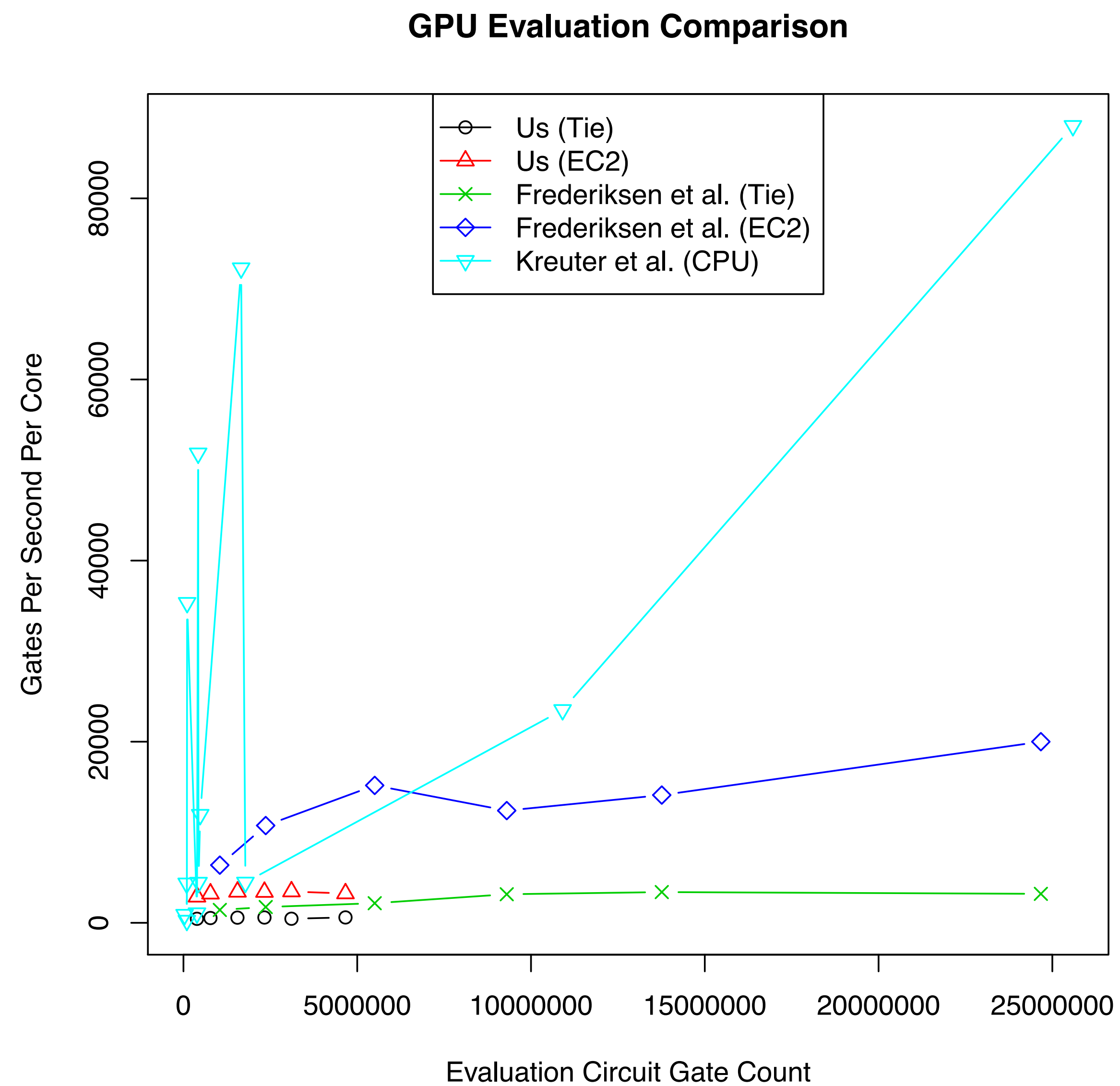


(a) Gates Gen vs. Time

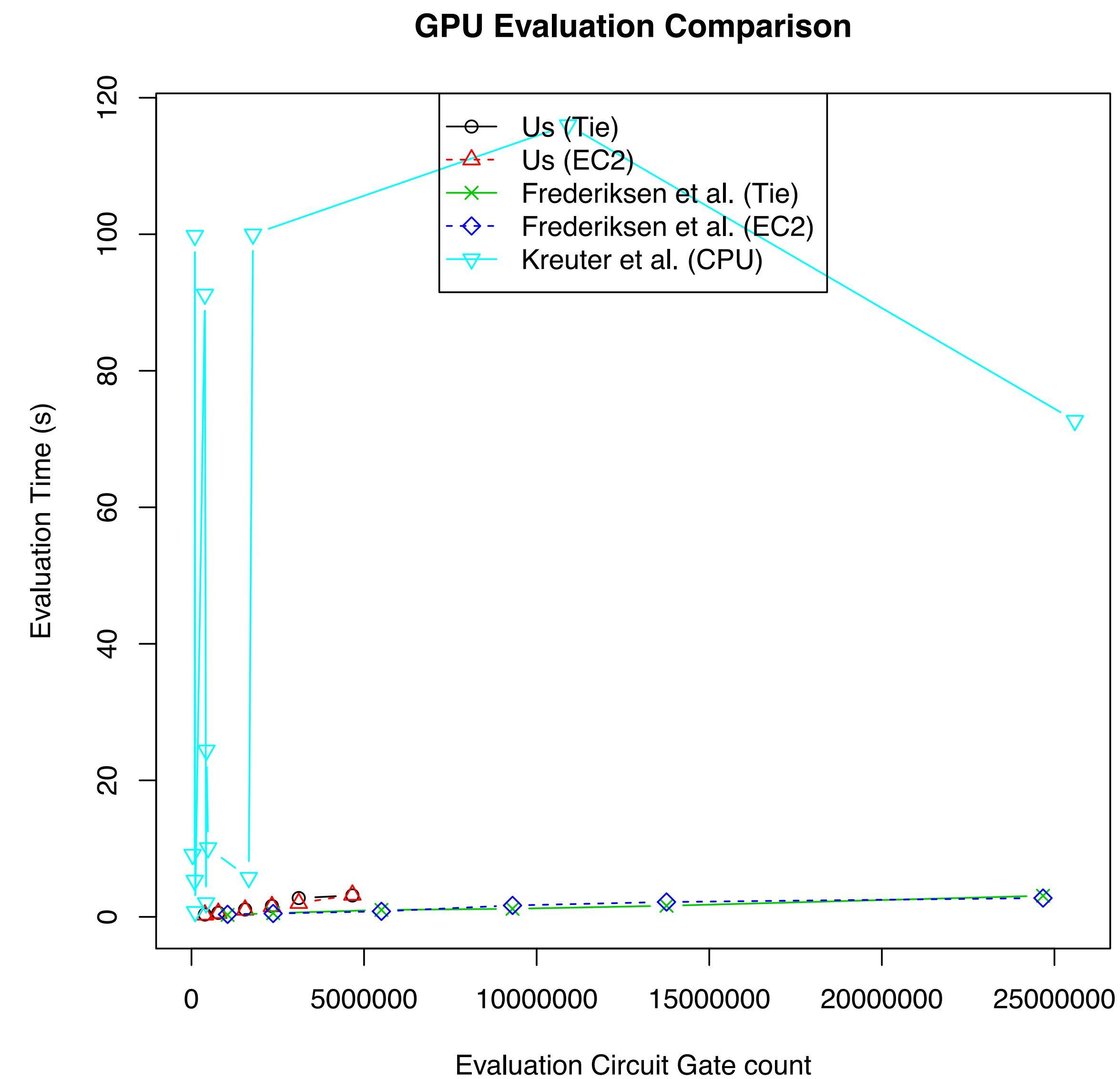


(b) Gate Gen Per Core Per Sec vs. Gate Count

Figure 2: Gate Generation Times comparing to Kreuter et al.[14].



(a) Comparing our GPU Eval Per Sec Per Core



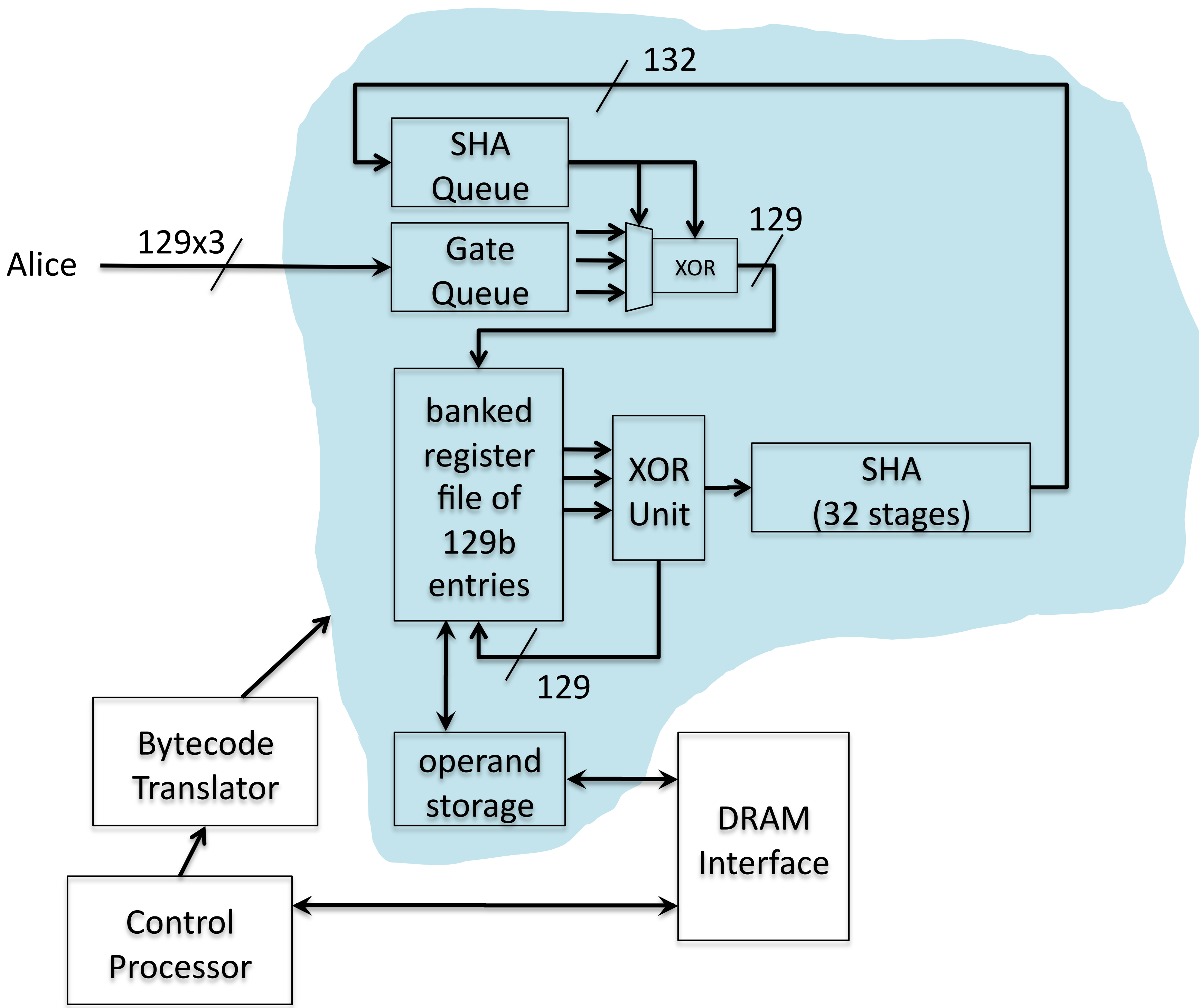
(b) Comparing our GPU Eval Overall

Figure 3: GPU Evaluation Times with comparison to Kreuter et al. [14], Frederiksen and Nielsen [5] and our GPU implementation.

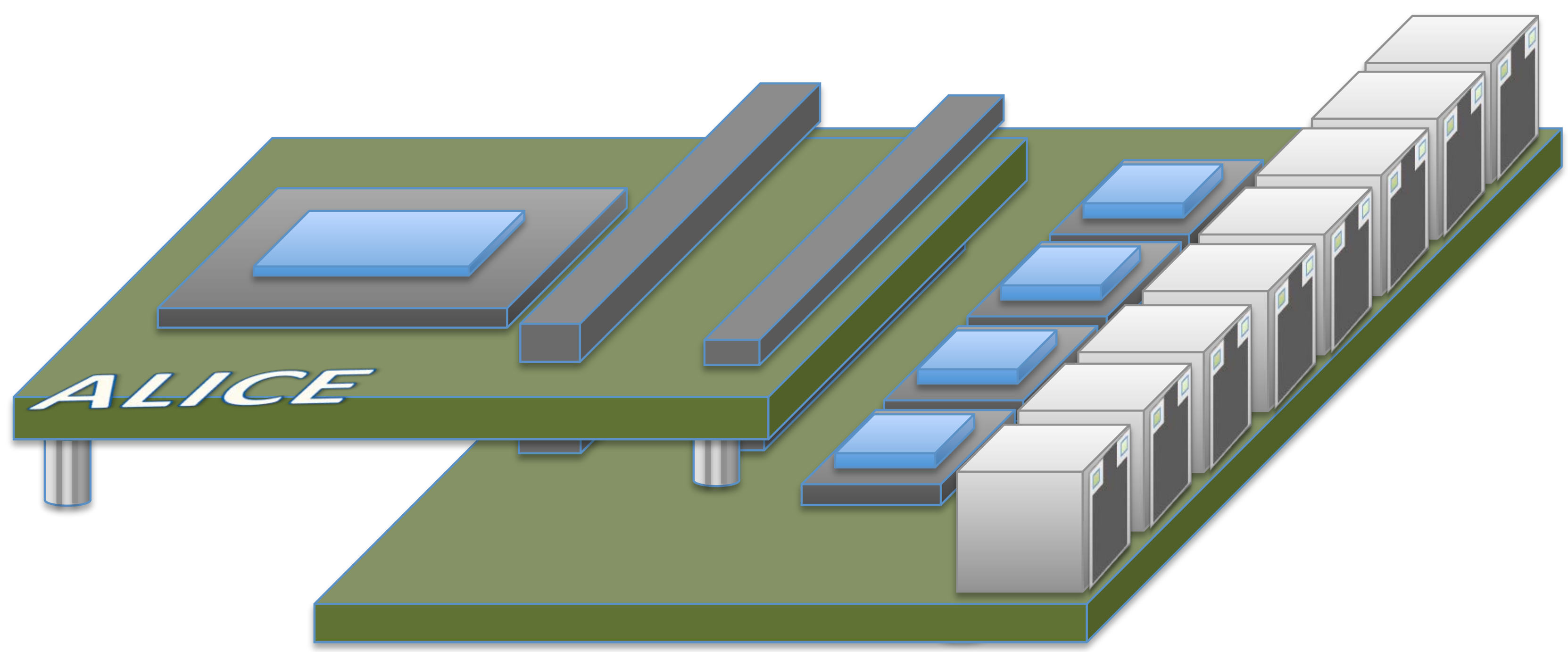
Plans

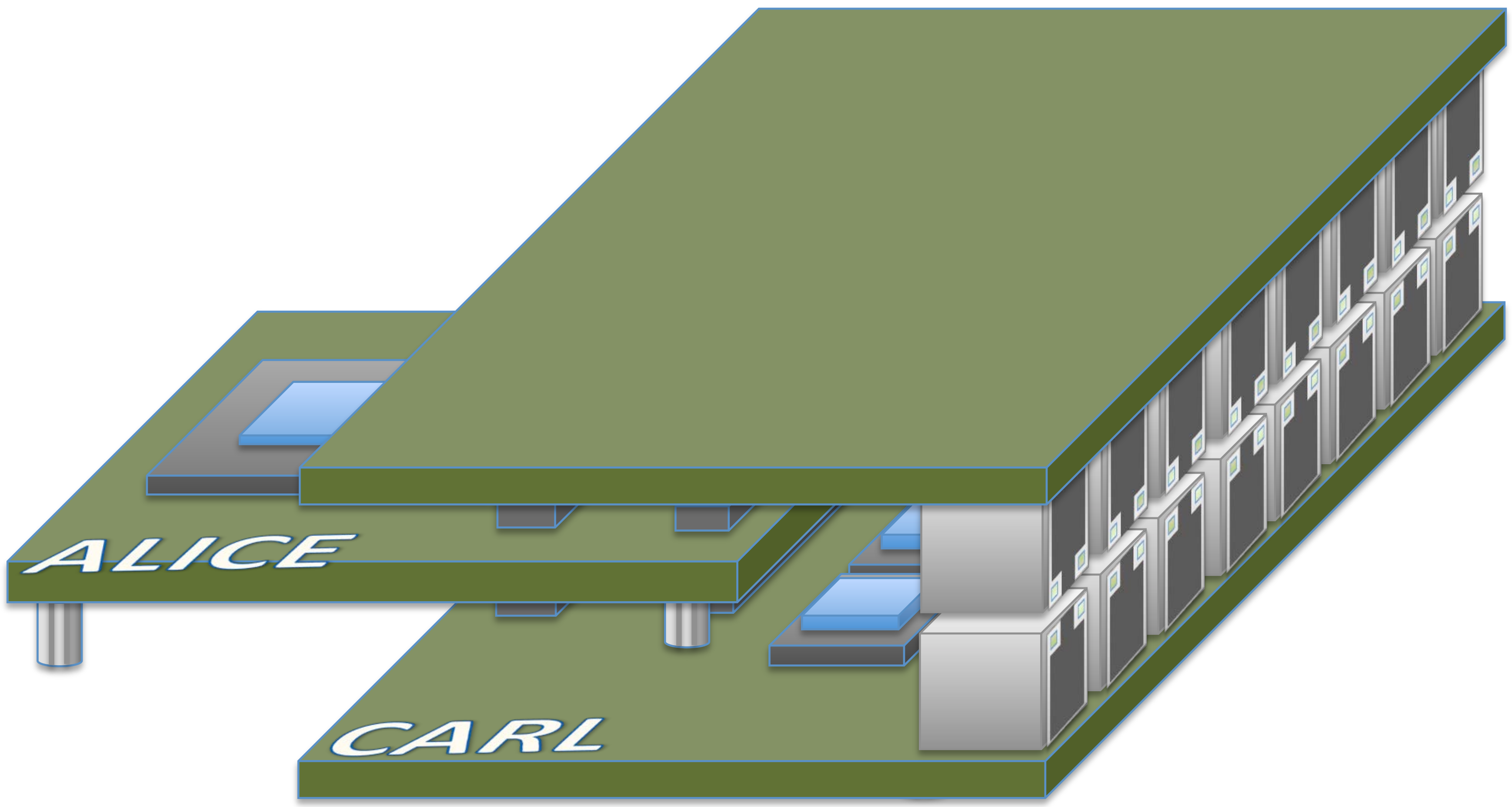


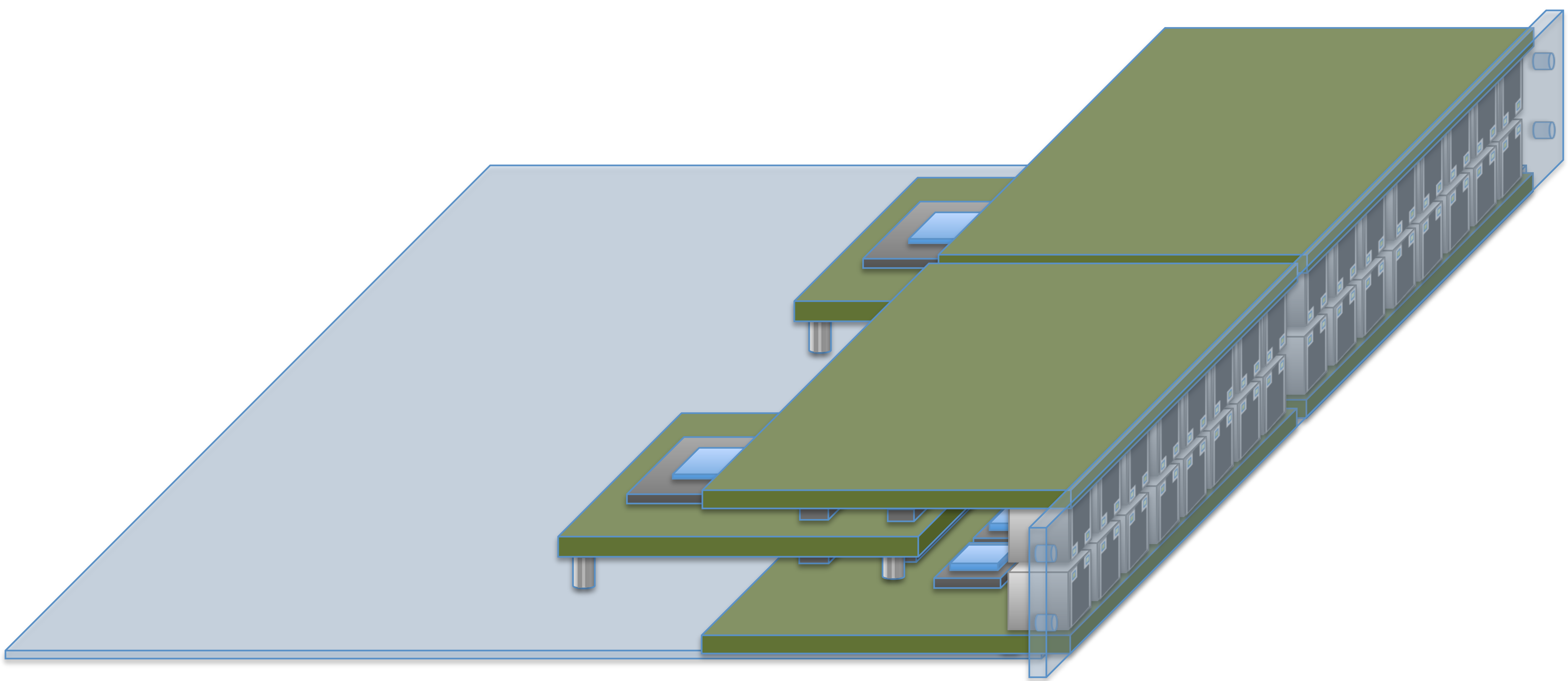


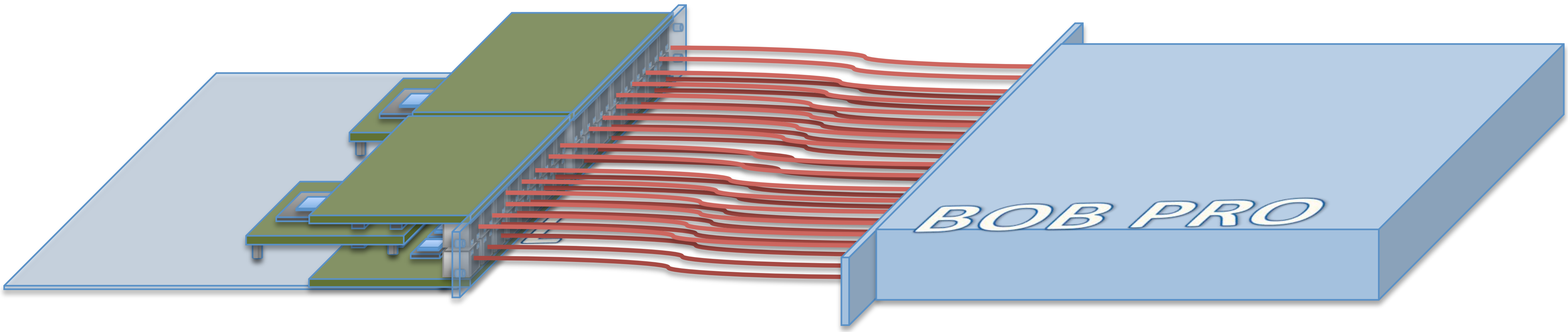



Michael Taylor


ALICE


ALICE
CARL




BOB PRO