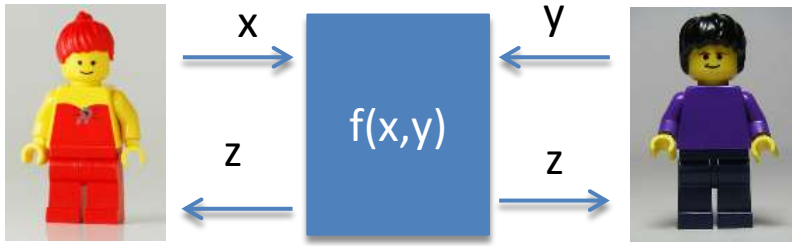
Bar-Ilan University

Hiding the Input Size in Secure Two-Party Computation

Yehuda Lindell, Kobbi Nissim, **Claudio Orlandi**

Secure Computation



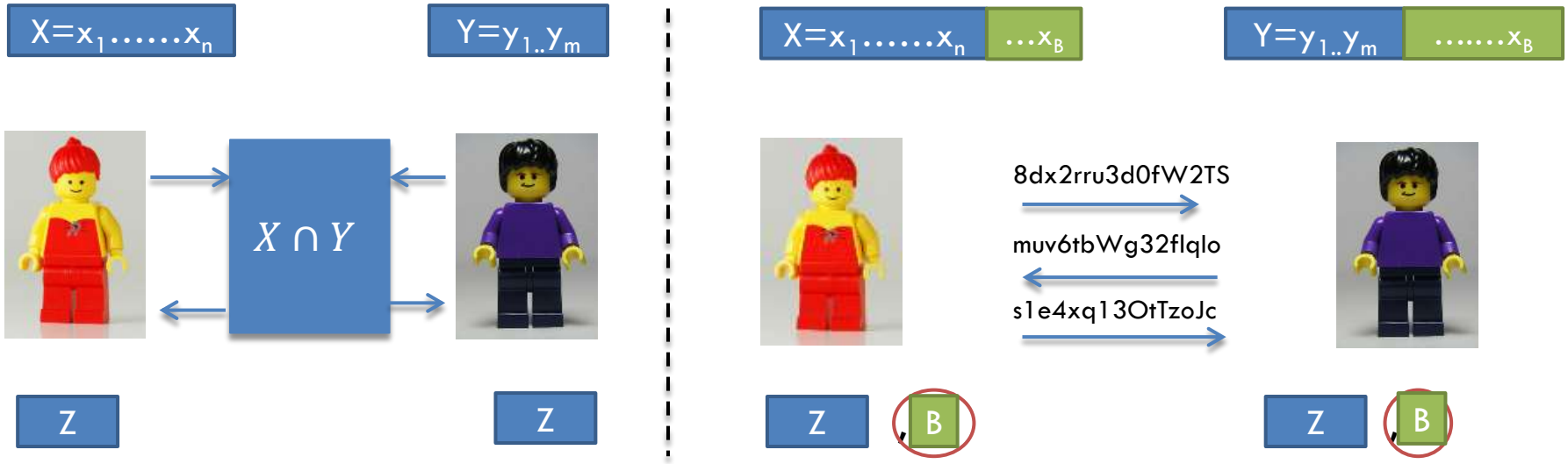
Trusted Party

Cryptographic Protocol

- Privacy
- Correctness
- Input Independence
- "The protocol is as secure as the ideal world".

Or is it?

Size matters!



- Private Set Intersection: the size of a list might be confidential
- Padding?
  - Just add a lot of "fake entries" to your DB
  - Requires an upper bound ☹
  - Inherent inefficiency ☹

Related Work

- MicaliRabinKilian'03:
  - Zero Knowledge Sets
- IshaiPaskin'07:
  - Branching programs (implies PSI, server size is hidden).
- AggarwalMishraPinkas'10:
  - Computing median.
- AtenieseDeCristofaroTsudik'11:
  - Specific protocol for PSI, client size is hidden.

Outline

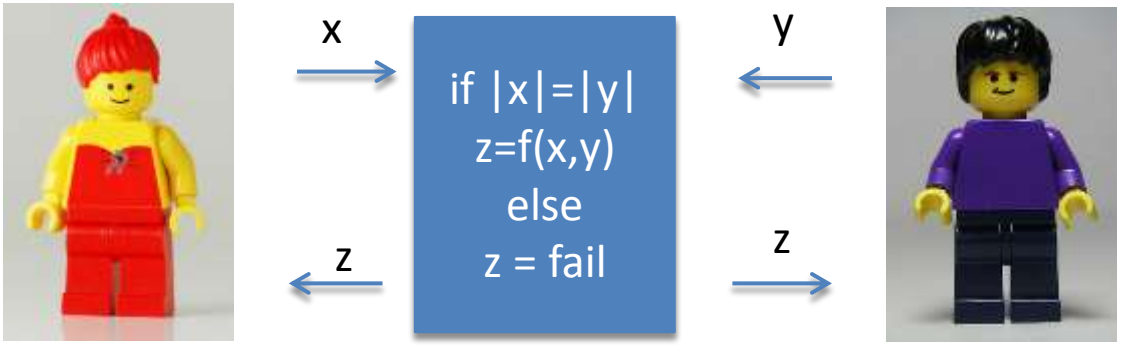
- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Dealing with input size

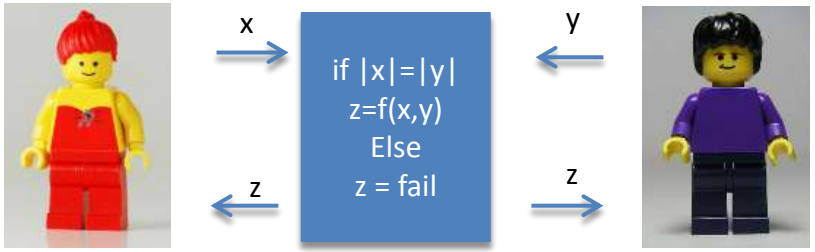
- Standard definition, e.g. [Gol04]



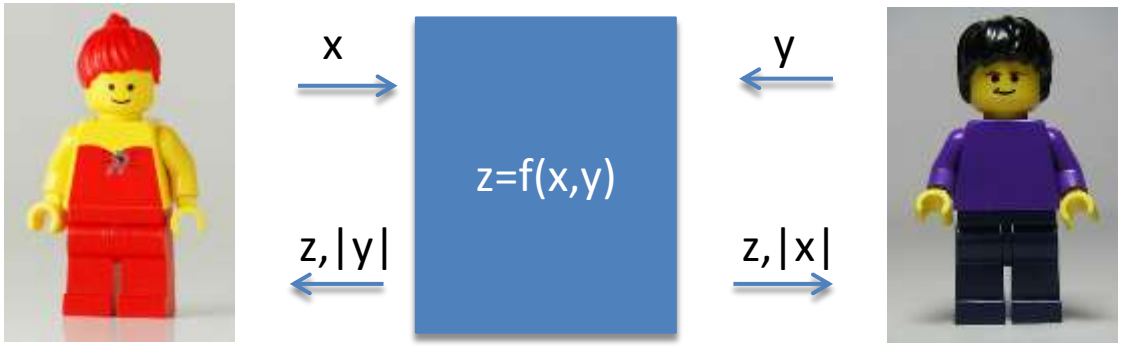
- Need to know other party's size in advance
  - (Also: input **size** independence?)

Dealing with input size

- Standard definition



- More Natural?

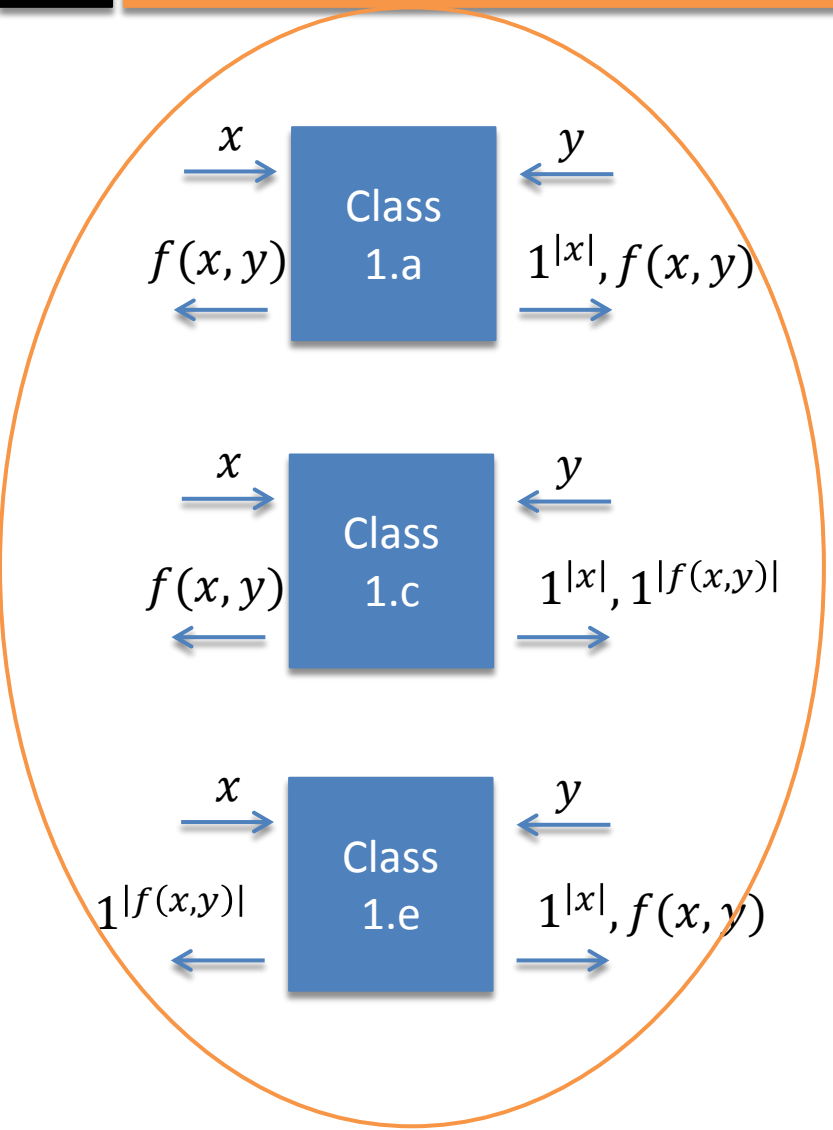


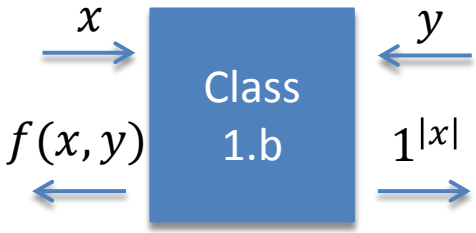
Ideal Model - Classes

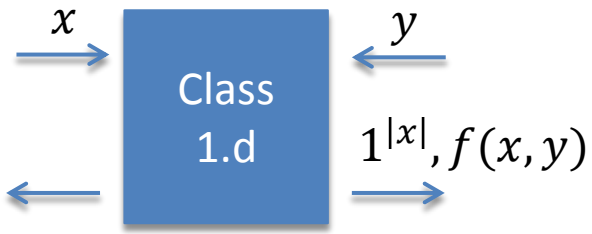
- Classes
  - 0: both input size are leaked
  - 1: Bob learns $|x|$, Alice does not learn $|y|$
  - 2: both input size are hidden
- Subclasses
  - Who gets output?
  - Is the output size leaked?
- Complete classification for symmetric functions $f(x, y) = f(y, x)$

Class 0

Class 1

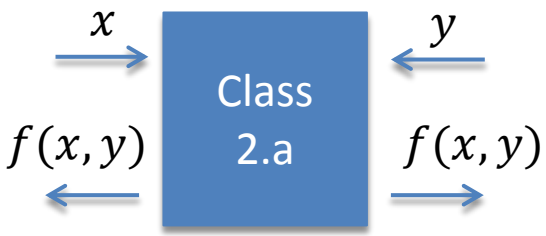


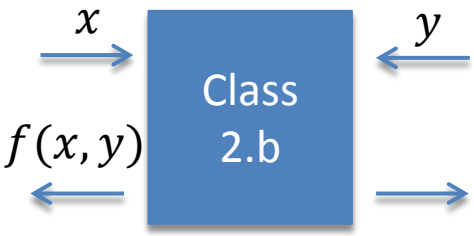


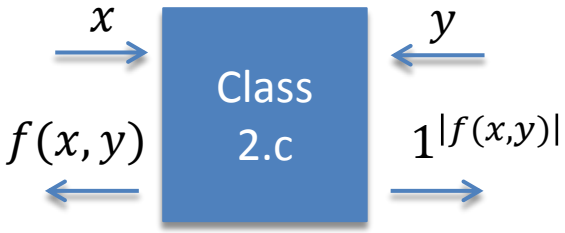


Essentially equivalent classes (outputs have same length)

Class 2







Definitional Issues

- (Std.) poly-time $= poly(x, k)$
- But here $|f(x,y)|$ is not bounded by $poly(x,k)$
- How to define poly-time?
  - **Vs. semi-honest:** running-time is polynomial in the lengths of input, output and security parameter.
- Security definition: quantify the size of the inputs at the end

Definition 2.2 (Security for Class $A.b$ – Semi-Honest) *Let $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a functionality, and let π be a polynomial time protocol for class $A.b$. We say that π* securely computes *f* in class *$A.b$* in the presence of semi-honest adversaries *if there exist probabilistic polynomial time-algorithms $\mathcal{S}_1, \mathcal{S}_2$ such that for every pair of polynomials $q_1(\cdot)$ and $q_2(\cdot)$,*

$$\left\{\left(\mathcal{S}_1(x, \text{OUTPUT}_1^{A.b}(x,y)), \text{OUTPUT}^{A.b}(x,y)\right)\right\}_{\kappa,x,y} \overset{c}{\equiv} \left\{(\text{view}_1^{\pi}(x,y,\kappa), \text{OUTPUT}^{\pi}(x,y,\kappa))\right\}_{\kappa,x,y}$$

$$\left\{\left(\mathcal{S}_2(y, \text{OUTPUT}_2^{A.b}(x,y)), \text{OUTPUT}^{A.b}(x,y)\right)\right\}_{\kappa,x,y} \overset{c}{\equiv} \left\{(\text{view}_2^{\pi}(x,y,\kappa), \text{OUTPUT}^{\pi}(x,y,\kappa))\right\}_{\kappa,x,y}$$

where $\kappa \in \mathbb{N}$, $x \in \{0,1\}^{q_1(\kappa)}$ and $y \in \{0,1\}^{q_2(\kappa)}$.

Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Tools

- Fully Homomorphic Encryption
  - $(G, E, D, Eval)$

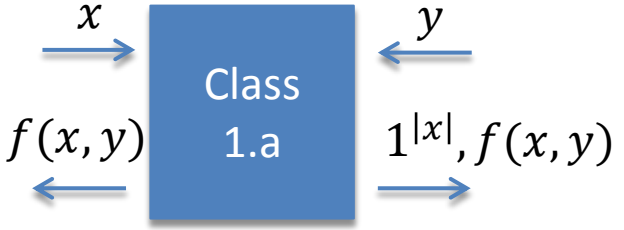
  - Correctness: (ewnp)

$$D_{sk}(Eval_{pk}\left(f, E_{pk}(x), E_{pk}(y)\right) = f(x, y)$$

  - Circuit privacy:

$$Eval_{pk}\left(f, E_{pk}(x), E_{pk}(y)\right) \approx E_{pk}(f(x, y))$$

Class 1.a



$(pk, sk) \leftarrow Gen(1^k)$

$c_x \leftarrow Enc_{pk}(x)$

pk, c_x →

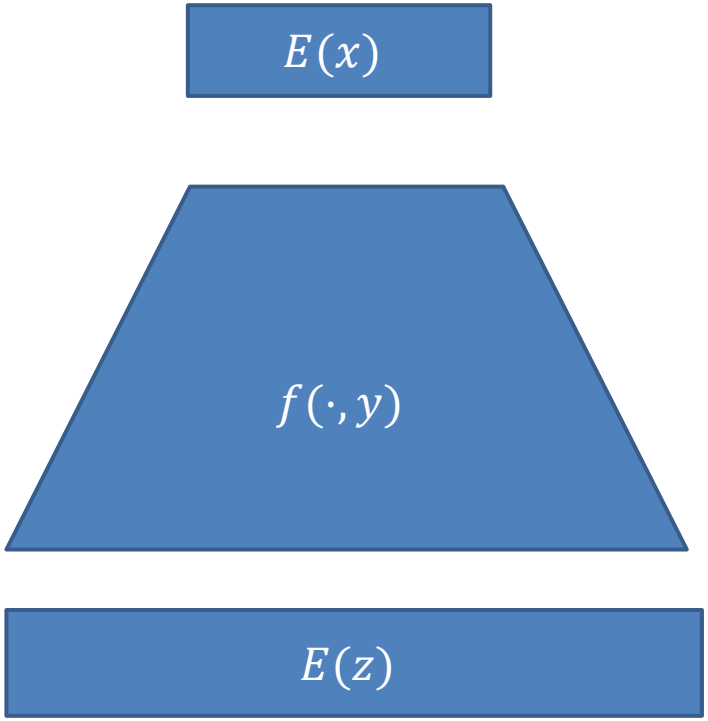
c_z ←

$z = Dec_{sk}(c_z)$

z →

$c_z = Eval_{pk}(f(\cdot, y), c)$

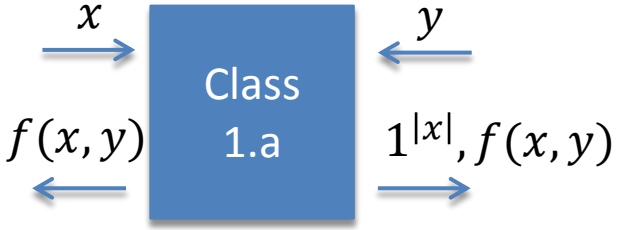
How big should the output be?



e.g. $z = x \cup y$

Clear that $|z| \leq |x| + |y|$
But how long exactly?
Any upper bound reveals info about $|y|$

Class 1.a



$(pk, sk) \leftarrow Gen(1^k)$

$c_x \leftarrow Enc_{pk}(x)$

pk, c_x →

c_ℓ ←

$c_\ell = Eval_{pk}(\mathbf{sizeof}(f\ (\cdot, y)), c)$

$\ell = Dec_{sk}\ (c_\ell)$

ℓ →

c_z ←

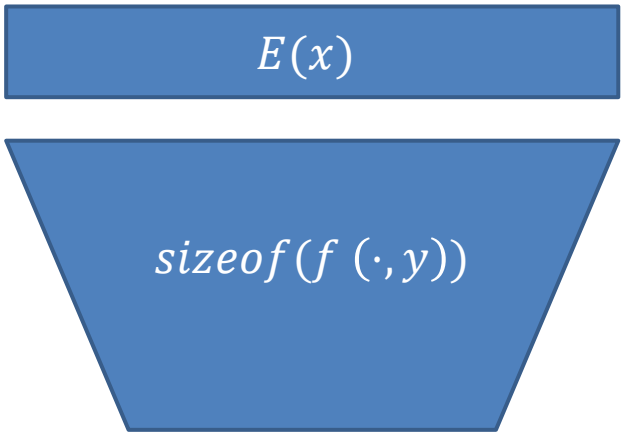
$c_z = Eval_{pk}(\boldsymbol{f}_\ell(\cdot, y), c)$

$z = Dec_{sk}\ (c_f)$

z →

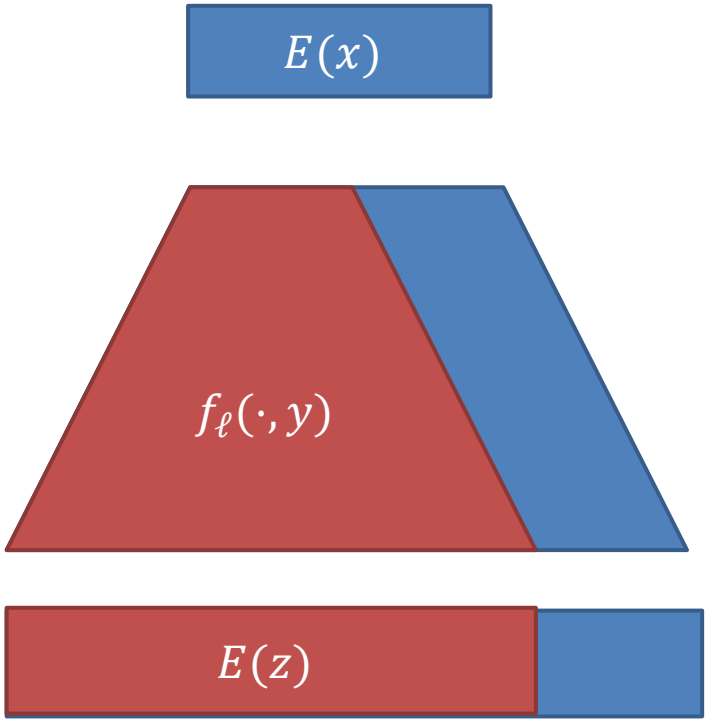
- Thm: FHE → $\forall f$ can be securely computed in Classes 1.a/c/e

How big should the output be?



Send to Alice

Alice opens $\ell = |z|$



Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Class 2

- Thm (informal): (Assuming FHE)
  - if f admits a *size-independent protocol*, then f can be computed in Class 2.a

- Proof idea:
  - compile the (insecure) communication efficient protocol into a secure one using FHE

Size Independent Protocols

- π is size independent for f if
  - Correct (except for negl(k))
  - Computation efficient (runtime $poly(input+k)$)
  - Communication efficient (bounded by $poly(k)$)

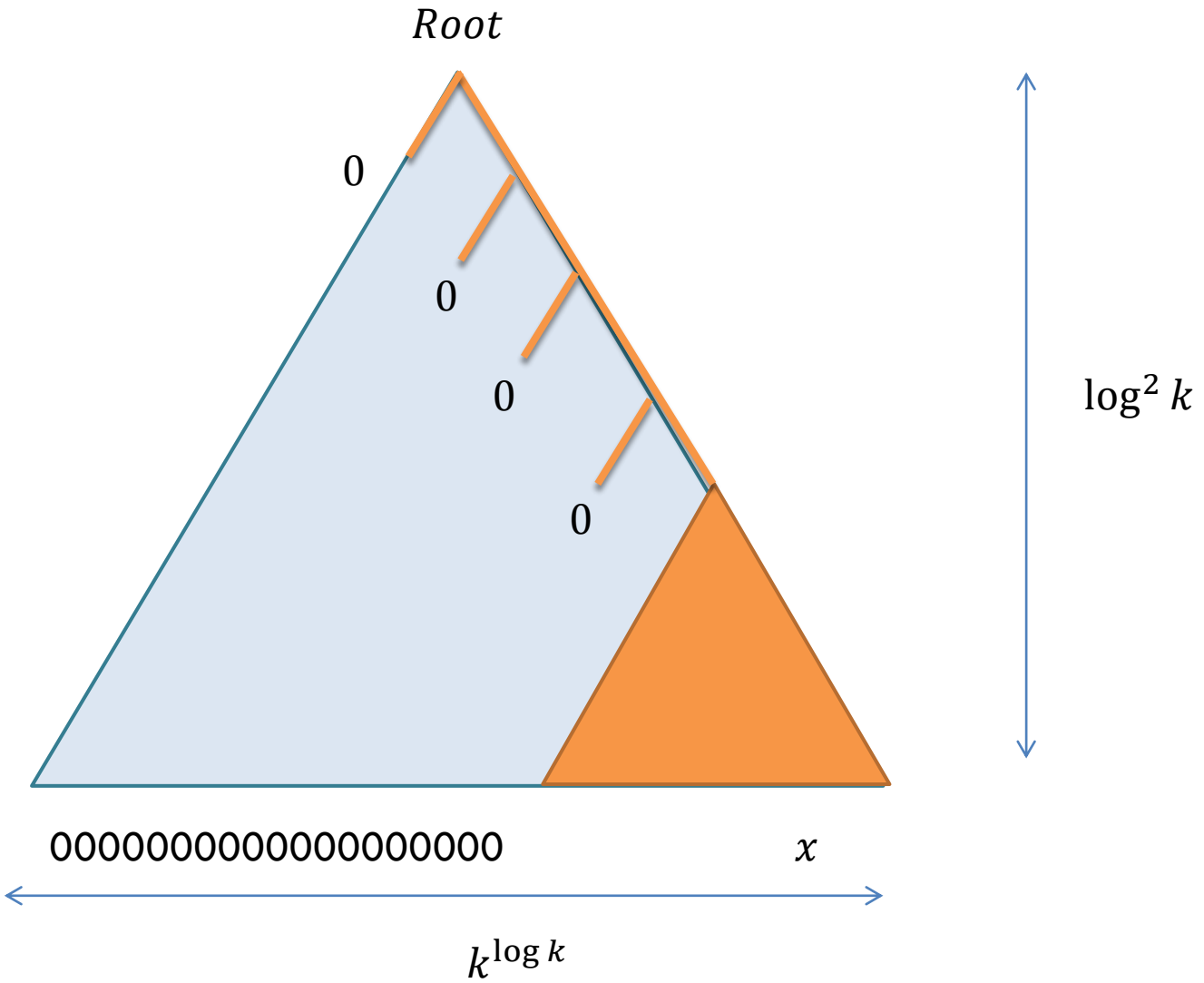
- (no "security" so far)

Example: Size-Independent protocol for Millionaire

- Tools:
  - Let $H : \{0,1\}^{2k} \rightarrow \{0,1\}^{k}$ s.t. $H(0,0) = 0$
  - $Tree(x) = \begin{matrix} x \; if \; |x| = k, else \\ H(Tree(x_L), Tree(x_R)) \end{matrix}$

Can compute Merkle Tree of depth $\log^2 k$ in time $poly(k)$

Merkle Tree



Not secure!!!

Size-Independent Millionaire's Protocol

$x = (x_L, x_R)$

$root \leftarrow Tree(x_L)$

$root_x$ →

$y = (y_L, y_R)$

$if\ root = Tree(y_L)$

$z \leftarrow R$, else, z $\leftarrow L$

← z

$x \leftarrow x_z$

$y \leftarrow y_z$

Until $|x| > k$

$|x| = k$

x →

Output z $\leftarrow (x > y)$

Summary

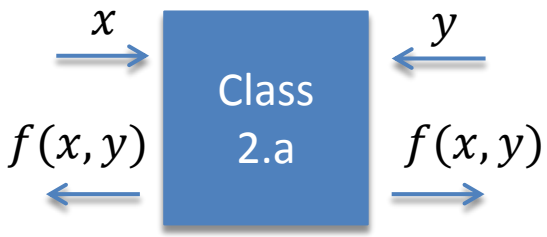
- Take size-independent protocol
  - (like the one just seen)
- Compile using FHE
  - (similar to Class 1 protocol)

➔ 2 Size-Hiding protocol

Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Lower Bounds



- There are functions that cannot be computed while hiding both parties' input size.
  - (Not everything can be computed in Class 2)

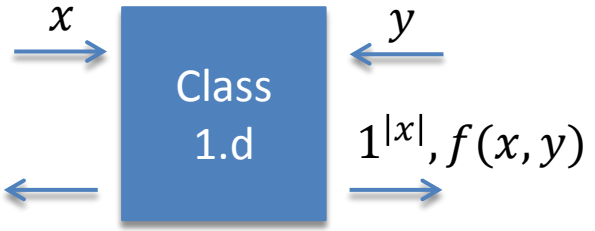
- Proof idea:
  - $IP(x,y)$ has comm. complexity $O(\min(|x|,|y|))$
  - Size Hiding IP **must** have comm. complexity $poly(k)$
  - Contradiction!

- (Also: Intersection, Hamming distance, etc.)

Class 1.b

- Size-hiding OT:
  - x = selection bit
  - $y = (y_0, y_1)$ two strings of different length
  - $f(x,y) = y_x$
- Thm: OT cannot be computed in Class 1.b
- Proof idea:
  - Transcript are independent of y_{1-x} , (security of sender)
  - Also independent of x, (security of receiver)
  - **must** be poly(k)
  - But! OT can be used to send more than $poly(k)$ bits.
  - Contradiction!

Class 1.d



- Oblivious multipoint PRF
  - $x =$ a PRF key
  - $y = (y_0, \ldots, y_n)$
  - $f(x,y) = (PRF_x(y_0), \ldots, PRF_x(y_n))$
- Thm: OMPRF cannot be computed in Class 1.d
- Proof idea:
  - Transcript must be independent of $|y|$
  - Simulator needs to "compress" the output.
  - PRF is indistinguishable from random function.
  - Simulator cannot compress random data.

Summary of Feasibility

	All f (bounded output)	All f (even unbounded output)	GT ($x > y$)	vecxor	Intersection	OT	omprf
2.a	×	×	✓	✓	×	✓	✓
2.b	×	×	✓	×	×	×	✓
2.c	×	×	✓	✓	×	✓	✓
1.a	✓	✓	✓	✓	✓	✓	✓
1.b	✓	×	✓	✓	✓	×	✓
1.c	✓	✓	✓	✓	✓	✓	✓
1.d	✓	×	✓	✓	✓	✓	×
1.e	✓	✓	✓	✓	✓	✓	✓

Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Dealing with Malicious Adversaries

- Definition?
  - For semi-honest: poly-time in input/output
  - For malicious: inputs/outputs are not well defined!

    → Protocol is poly time if honest party run in poly(adversary runtime)

  - Inherent "DoS"

Size-hiding GMW?

- Standard ZK reveals witness size
- Universal argument + FHE → Size-hiding ZK
- But it has only "weak" proof of knowledge!
  - Simulator can extract every bit from the input in poly-time.
- Fix: add an "oblivious proof of work"
  - Can be proven secure under exact birthday paradox assumption.

Outline

- Definition and Classification
- Feasibility
  - 1-size hiding
  - 2-size hiding
  - Negative Results
- Malicious Security
- Conclusions and Open Problems

Conclusions

- Hiding the input size is (sometimes) possible.
  - Don't give up!

- Open problems:
  - Efficient protocols for specific tasks
  - Malicious security under standard assumption?
  - …