Quantum query complexity and the adversary bound
Part I: The adversary bound

Alexander Belov
University of Latvia

22nd EWSCS, 5-10 March 2017, Palmse

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Introduction

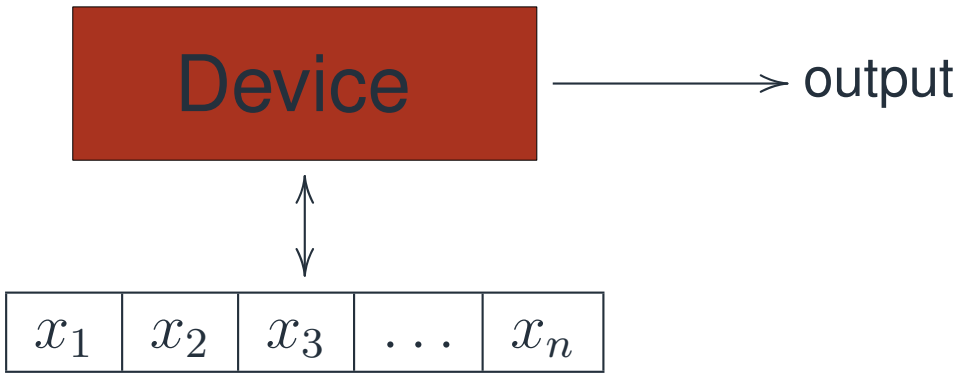
Settings

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

A computational problem:

$$f : [q]^n \supseteq \mathcal{D} \to \{0, 1\}$$

A computational device (deterministic, randomised, or quantum) :



- Query complexity: number of queries to the input string (worst-case) required to solve the problem.
- Both upper and lower bounds.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why quantum query complexity?

- Why quantum **query** complexity?
- Why **quantum** query complexity?
- Why **quantum query** complexity?

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why query complexity?

Reason 1: For some problems, this is the right model.

- In hypothesis testing, we are interested in reducing the number of experiments. Experiments are expensive, computation is not so.
- In property testing, complexity is traditionally measured in the number of samples.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why **query** complexity?

Reason 1: For some problems, this is the right model.

Reason 2: Because we can.

- We are interested in time complexity usually, but in most cases we can only prove lower bounds by proving lower bound on query complexity.
- In cryptography, proofs in the random oracle model are, essentially, query complexity lower bounds.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why **query** complexity?

Reason 1: For some problems, this is the right model.
Reason 2: Because we can.
Reason 3: Query algorithm can give insights into the problem.

- Trying to minimise query complexity of an algorithms result in our better understanding of algorithms and the problem.
- One might hope to implement query-efficient algorithm time-efficiently.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why **query** complexity?

Reason 1: For some problems, this is the right model.
Reason 2: Because we can.
Reason 3: Query algorithm can give insights into the problem.
Reason 4: Query complexity gives impossibility results.

- We can prove that black-box approaches are doomed.
- We can exclude a lower bound via query complexity.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why **quantum** query complexity?

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why quantum query complexity?

Reason 1: You do not need to understand quantum computation to study quantum query complexity.

- Quantum query complexity is tightly characterised by a semi-definite optimisation problem: the adversary bound.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why quantum query complexity?

Reason 1: You do not need to understand quantum computation to study quantum query complexity.

Reason 2: In some sense, we understand quantum query complexity better than randomised query complexity.

- Randomised query complexity is a collection of separated results.
- Quantum query complexity start resembling a theory.

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

Why quantum query complexity?

Reason 1: You do not need to understand quantum computation to study quantum query complexity.

Reason 2: In some sense, we understand quantum query complexity better than randomised query complexity.

Reason 3: Quantum query complexity provides upper bound on approximate polynomial degree.

- Approximating polynomials are used in various fields (e.g., learning theory)
- One can get an approximating polynomial by constructing a quantum query algorithm.

Short outline

Introduction
Settings
Why?
Sub-question 1
Sub-question 2
Sub-question 3
Short outline
Basic Adversary
Spectral Adversary
Dual Adversary
Composition

We are mostly considering total functions

$$f : [q]^n \to \{0, 1\}.$$

- For such functions, one can only get a polynomial improvement (at most 6-th power)
- We will mostly consider sub-quadratic improvements.

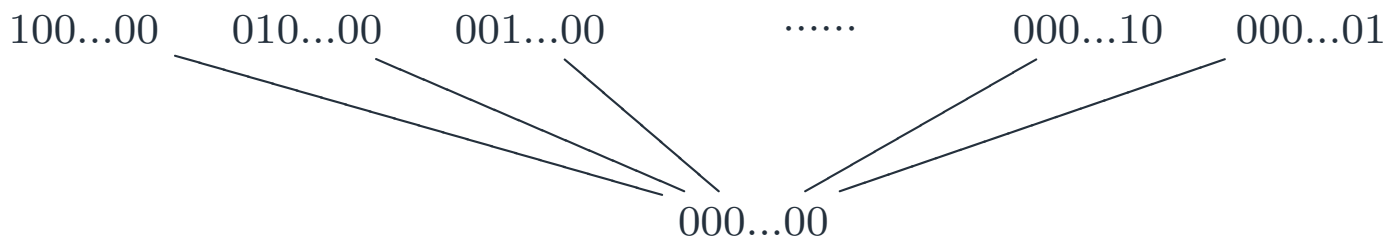
Introduction
Basic Adversary
Main idea
Quantum Version
k-threshold function
Spectral Adversary
Dual Adversary
Composition

Basic Adversary

Introduction
Basic Adversary
Main idea
Quantum Version
k-threshold function
Spectral Adversary
Dual Adversary
Composition

Distinguishing inputs

OR function:



No randomised algorithm can distinguish all these pairs of inputs in $o(n)$ queries.
Hence, randomised query complexity is $\Omega(n)$.

Quantum Version

Introduction

Basic Adversary

Main idea

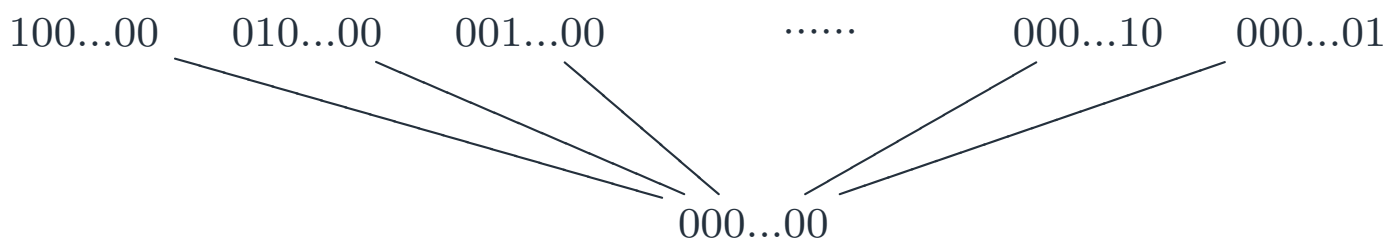
Quantum Version

k-threshold function

Spectral Adversary

Dual Adversary

Composition



Select:

$X \subseteq f^{-1}(1)$

$Y \subseteq f^{-1}(0)$

Relation $\sim$ between X and Y

Quantum Version

Introduction

Basic Adversary

Main idea

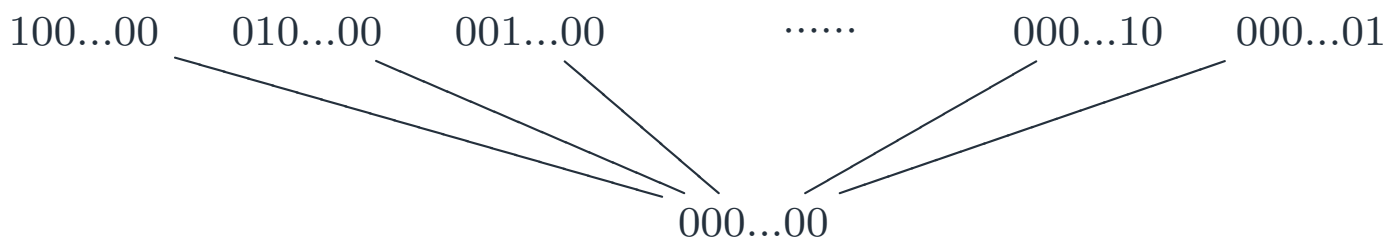
Quantum Version

k-threshold function

Spectral Adversary

Dual Adversary

Composition



Calculate:

m : minimum, over $x \in X$, of the number of y with $x \sim y$: 1

m' : minimum, over $y \in Y$, of the number of x with $x \sim y$: n

$\ell_{x,j}$: number of $y \in Y$ such that $x \sim y$ and $x_j \neq y_j$: 1

$\ell'_{y,j}$: number of $x \in X$ such that $x \sim y$ and $x_j \neq y_j$: 1

$\ell_{\max}$: maximum of $\ell_{x,j}\ell'_{y,j}$ over $x \sim y$ and j such that $x_j \neq y_j$: 1

$$Q(f) = \Omega\left(\sqrt{\frac{mm'}{\ell_{\max}}}\right) = \Omega(\sqrt{n})$$

k-threshold function

Introduction
Basic Adversary
Main idea
Quantum Version
k-threshold function
Spectral Adversary
Dual Adversary
Composition

k-threshold function:

$f(x) = 1$ if there are at least k ones in the input.

$X \subseteq f^{-1}(1)$: x of Hamming weight k
$Y \subseteq f^{-1}(0)$: y of Hamming weight $k-1$
$\sim$: $x \sim y$ if x and y differ in 1 position

$$x: \quad 111\ldots11100\ldots00$$
$$y: \quad 111\ldots11000\ldots00$$

k-threshold function

Introduction

Basic Adversary

Main idea

Quantum Version

k-threshold function

Spectral Adversary

Dual Adversary

Composition

$$x: \quad 111\ldots11100\ldots00$$
$$y: \quad 111\ldots11000\ldots00$$

m : minimum, over $x \in X$, of number y with $x \sim y$: k

m' : minimum, over $y \in Y$, of number x with $x \sim y$: $n-k+1$

$\ell_{x,j}$: number of $y \in Y$, such that $x \sim y$ and $x_j \neq y_j$: 1

$\ell'_{y,j}$: number of $x \in X$, such that $x \sim y$ and $x_j \neq y_j$: 1

$\ell_{\max}$: maximum of $\ell_{x,j}\ell'_{y,j}$ over $x \sim y$ and j such that $x_j \neq y_j$: 1

$$Q(k\text{-threshold}) = \Omega\left(\sqrt{\frac{mm'}{\ell_{\max}}}\right) = \Omega\left(\sqrt{k(n-k+1)}\right)$$

This is tight!

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

Spectral Adversary

Matrix Representation

Introduction

Basic Adversary

Spectral Adversary

Matrix Representation

Example

Theorem

Proof

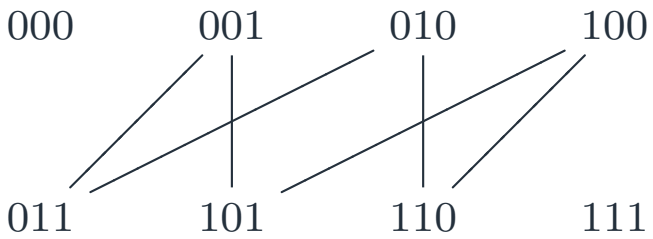
Spectral Adversary

Flavours

Dual Adversary

Composition

We had a relation $x \sim y$



Represent as a 01-matrix:

$$\Gamma = \begin{array}{c|cccc} & 011 & 101 & 110 & 111 \\ \hline 000 & 0 & 0 & 0 & 0 \\ 001 & 1 & 1 & 0 & 0 \\ 010 & 1 & 0 & 1 & 0 \\ 100 & 0 & 1 & 1 & 0 \end{array}$$

Matrix Representation

Introduction

Basic Adversary

Spectral Adversary

Matrix Representation

Example

Theorem

Proof

Spectral Adversary

Flavours

Dual Adversary

Composition

We had

$\ell_{x,j}$: number of $y \in Y$, such that $x \sim y$ and $x_j \neq y_j$

$\ell'_{y,j}$: number of $x \in X$, such that $x \sim y$ and $x_j \neq y_j$

$\ell_{\max}$: maximum of $\ell_{x,j}\ell'_{y,j}$ over $x \sim y$ and j such that $x_j \neq y_j$

Erase all entries (x, y) such that $x_j = y_j$:

$\Gamma \circ \Delta_1 =$

	011	101	110	111
000	0	0	0	0
001	0	1	0	0
010	0	0	1	0
100	0	0	0	0

Matrix representation

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

We had

$$Q(f) = \Omega\left(\sqrt{\frac{mm'}{\ell_{\max}}}\right)$$

Spectral version of the adversary bound

$$\begin{aligned} &\text{maximise} && \|\Gamma\| \\ &\text{subject to} && \|\Gamma \circ \Delta_j\| \leq 1 && \text{for all } j \in [n]; \end{aligned}$$

Where $\|\cdot\|$ is the spectral norm:

$$\|\Gamma\| = \max_{\|u\|=1,\|v\|=1} |u^* \Gamma v|$$

Example: OR function

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

maximise $\|\Gamma\|$

subject to $\|\Gamma \circ \Delta_j\| \leq 1$ for all $j \in [n]$;

$$\Gamma = \begin{array}{c|c} & 00\ldots0 \\ \hline 10\ldots00 & 1 \\ 01\ldots00 & 1 \\ \vdots & \vdots \\ 00\ldots01 & 1 \end{array} \qquad \Gamma \circ \Delta_2 = \begin{array}{c|c} & 00\ldots0 \\ \hline 10\ldots00 & 0 \\ 01\ldots00 & 1 \\ \vdots & \vdots \\ 00\ldots01 & 0 \end{array}$$

$$\|\Gamma\| = \sqrt{n} \qquad \|\Gamma \circ \Delta_j\| = 1$$

Theorem

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

maximise	$\|\Gamma\|$	
subject to	$\|\Gamma \circ \Delta_j\| \leq 1$	for all $j \in [n]$;

implies

m : minimum, over $x \in X$, of number y with $x \sim y$
m' : minimum, over $y \in Y$, of number x with $x \sim y$
$\ell_{x,j}$: number of $y \in Y$, such that $x \sim y$ and $x_j \neq y_j$
$\ell'_{y,j}$: number of $x \in X$, such that $x \sim y$ and $x_j \neq y_j$
$\ell_{\max}$: maximum of $\ell_{x,j}\ell'_{y,j}$ over $x \sim y$ and j such that $x_j \neq y_j$

$$Q(f) = \Omega\left(\sqrt{\frac{mm'}{\ell_{\max}}}\right)$$

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

Lemma

Assume A, B and C are real matrices such that $A = B \circ C$.
Then,

$$\|A\| \leq \max_{i,j\,:\,A[\![i,j]\!]\neq 0} r_i(B)c_j(C),$$

where
$r_i(B)$ is the ℓ_2-norm of the i-th row of B, and
$c_j(C)$ is the ℓ_2-norm of the j-th column of C.

Spectral Adversary

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

maximise $\|\Gamma\|$

subject to $\|\Gamma \circ \Delta_j\| \leq 1$ for all $j \in [n]$;

works for any matrix

$\Gamma =$

	011	101	110	111
000	4.5	3.1	1.2	0.5
001	−2.1	7.8	2.5	6.9
010	9.1	−1.8	1.3	−0.2
100	−7.4	6.2	4.3	5.5

$\Gamma \circ \Delta_3 =$

	011	101	110	111
000	4.5	3.1	0	0.5
001	0	0	2.5	0
010	9.1	−1.8	0	−0.2
100	−7.4	6.2	0	5.5

Flavours

Introduction
Basic Adversary
Spectral Adversary
Matrix Representation
Example
Theorem
Proof
Spectral Adversary
Flavours
Dual Adversary
Composition

Basic adversary

Γ has only 0s and 1s.

Positive-weighted adversary

Γ has only non-negative entries.

- Combinatorial interpretation using weights and the Lemma.
- Subject to some limitations:
  certificate complexity and property testing barriers.

Negative-weighted adversary

Γ is arbitrary real matrix.

- No nice combinatorial interpretation is known.
- It is tight!

Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Duality
PSD matrices
Composition

Dual Adversary

Duality

Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Duality
PSD matrices
Composition

The adversary bound

maximise $\|\Gamma\|$

subject to $\|\Gamma \circ \Delta_j\| \leq 1$ for all $j \in [n]$;

has dual formulation

minimise $\max_z \sum_{j \in [n]} X_j[\![z, z]\!]$

subject to $\sum_{j : x_j \neq y_j} X_j[\![x, y]\!] = 1$ whenever $f(x) \neq f(y)$;

X_j is a p.s.d. $\mathcal{D} \times \mathcal{D}$ matrix for all $j \in [n]$,

Positive Semi-Definite Matrices

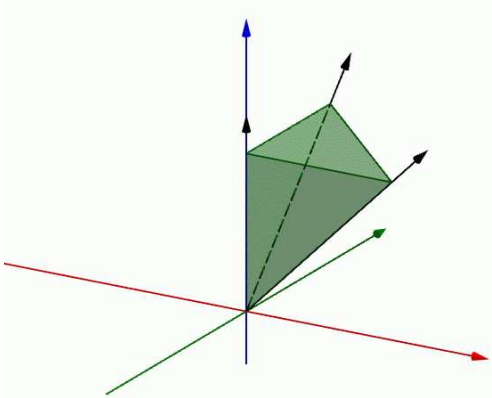
Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Duality
PSD matrices
Composition

Usual Definition: A complex matrix A is positive semi-definite iff

- it is Hermitian: $A^* = A$; and
- all its eigenvalues are non-negative real numbers.

Procedural Definition:

- Every matrix of the form uu^* is PSD (rank-1).
- Every linear combination of PSD matrices with **positive** coefficients is PSD.



Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Composition
Definition
Adversary Bound

Composition

Definition

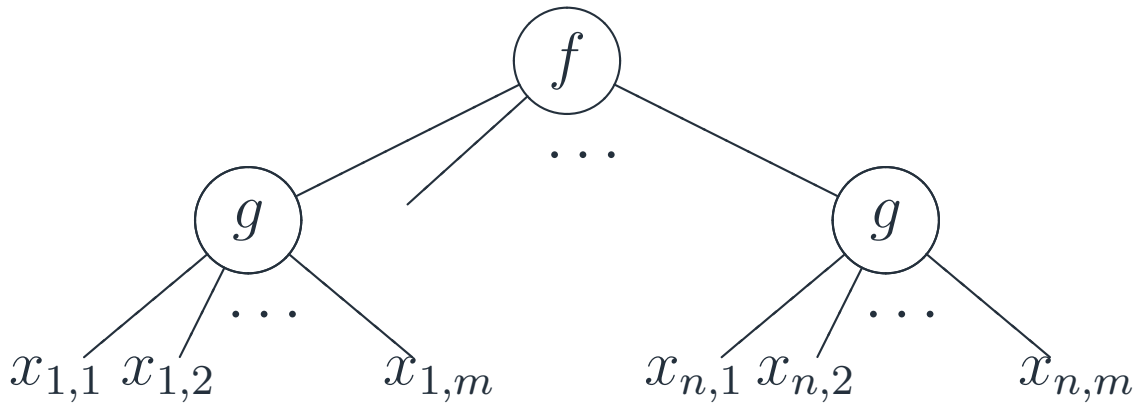
Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Composition
Definition
Adversary Bound

Let

$$f\colon \{0,1\}^n \to \{0,1\} \qquad \text{and} \qquad g\colon \{0,1\}^m \to \{0,1\}$$

be functions. Define the composed function

$$f\circ g(x_{1,1},\ldots,x_{n,m}) = f\big(g(x_{1,1},\ldots,x_{1,m}),\ldots,g(x_{n,1},\ldots,x_{n,m})\big)$$



Adversary Bound

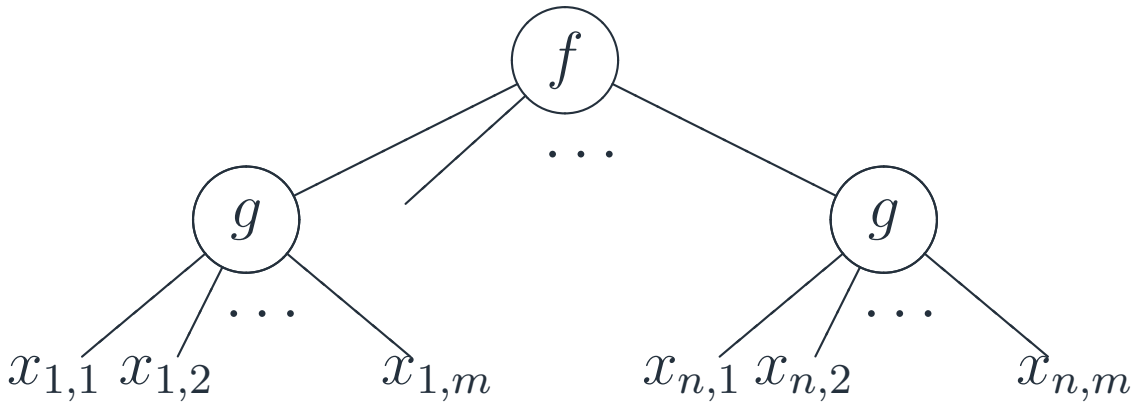
Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Composition
Definition
Adversary Bound

We have

$$\mathrm{Adv}^{\pm}(f \circ g) \leq \mathrm{Adv}^{\pm}(f)\,\mathrm{Adv}^{\pm}(g)$$

In particular,

$$Q(f \circ g) = O\big(Q(f)\,Q(g)\big)$$



- We save a logarithmic factor.
- On each composition!

Adversary Bound

Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Composition
Definition
Adversary Bound

We also have

$$\mathrm{Adv}^{\pm}(f \circ g) = \mathrm{Adv}^{\pm}(f)\,\mathrm{Adv}^{\pm}(g)$$

If $f^{(k)}$ is the k-th composition, then

$$Q(f^{(k)}) = \Theta(\mathrm{Adv}^{\pm}(f)^k)$$

Or,

$$\mathrm{Adv}^{\pm}(f) = \lim_{k\to\infty} \sqrt[k]{Q(f^{(k)})}.$$

Introduction
Basic Adversary
Spectral Adversary
Dual Adversary
Composition
Definition
Adversary Bound

Thank you!