Quantum Proofs for Classical Theorems

Ronald de Wolf

CWI

Universiteit van Amsterdam

Oxford, October 24, 2014

Unexpected proofs: Complex numbers

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

$\cos(x + y)$

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

$\cos(x + y) = \Re(e^{i(x+y)})$

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

$\cos(x + y) = \Re(e^{i(x+y)}) = \Re(e^{ix}e^{iy})$

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

$\cos(x+y) = \Re(e^{i(x+y)}) = \Re(e^{ix}e^{iy})$

$$= \Re(\cos(x)\cos(y) - \sin(x)\sin(y) + i\cos(x)\sin(y) + i\sin(x)\cos(y))$$

Unexpected proofs: Complex numbers

How to prove the following identity about *real* numbers

$\cos(x + y) = \cos(x)\cos(y) - \sin(x)\sin(y)$?

Go to *complex* numbers!

$e^{ix} = \cos(x) + i\sin(x)$

$\cos(x + y) = \Re(e^{i(x+y)}) = \Re(e^{ix}e^{iy})$

$= \Re(\cos(x)\cos(y) - \sin(x)\sin(y) +$
$\quad i\cos(x)\sin(y) + i\sin(x)\cos(y))$

$= \cos(x)\cos(y) - \sin(x)\sin(y)$

Unexpected proofs: Probabilities

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ “crosses” (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ "crosses" (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}]$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ "crosses" (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}]$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ ``crosses'' (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ "crosses" (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$
4. Expected number of crossing edges:

   $$\mathbb{E}\left[\sum_{(i,j)\in E} X_{ij}\right]$$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ "crosses" (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$
4. Expected number of crossing edges:

$$\mathbb{E}\left[\sum_{(i,j)\in E} X_{ij}\right] = \sum_{(i,j)\in E} \mathbb{E}[X_{ij}]$$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ "crosses" (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$
4. Expected number of crossing edges:

$$\mathbb{E}\left[\sum_{(i,j)\in E} X_{ij}\right] = \sum_{(i,j)\in E} \mathbb{E}[X_{ij}] = \sum_{(i,j)\in E} \frac{1}{2}$$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ “crosses” (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$
4. Expected number of crossing edges:

$$\mathbb{E}\left[\sum_{(i,j)\in E} X_{ij}\right] = \sum_{(i,j)\in E} \mathbb{E}[X_{ij}] = \sum_{(i,j)\in E} \frac{1}{2} = m/2$$

Unexpected proofs: Probabilities

Probabilistic method (Erdős, Alon & Spencer)

Theorem: Every graph (V, E) with m edges contains a bipartite subgraph with $m/2$ edges

Proof:

1. Pick vertex-set $T \subseteq V$ at random
2. Set $X_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ ``crosses'' (between } T \text{ and } \overline{T}) \\ 0 & \textit{otherwise} \end{cases}$
3. $\mathbb{E}[X_{ij}] = \Pr[\text{edge } (i,j) \text{ crosses}] = 1/2$
4. Expected number of crossing edges:
$$\mathbb{E}\left[\sum_{(i,j)\in E} X_{ij}\right] = \sum_{(i,j)\in E} \mathbb{E}[X_{ij}] = \sum_{(i,j)\in E} \frac{1}{2} = m/2$$
5. But then there is a T with at least $m/2$ crossing edges!

Unexpected proofs: Information theory

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:
  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \le n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:
  1. Def $S = \{x \in \{0,1\}^n : |x| \le d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \ldots X_n$ be uniformly random element of S

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \dots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \leq d/n$

Unexpected proofs: Information theory

- How much is $\displaystyle\sum_{i=0}^{d}\binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:
  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d}\binom{n}{i}$
  2. Let $X = X_1 \ldots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \leq d/n$, so $H(X_i) \leq H(d/n)$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \le n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

  1. Def $S = \{x \in \{0,1\}^n : |x| \le d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \ldots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \le d/n$, so $H(X_i) \le H(d/n)$
  4. $\log |S|$

Unexpected proofs: Information theory

- How much is $\displaystyle\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \dots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \leq d/n$, so $H(X_i) \leq H(d/n)$
  4. $\log |S| = H(X)$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \ldots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \leq d/n$, so $H(X_i) \leq H(d/n)$
  4. $\log|S| = H(X) \leq \sum_{i=1}^{n} H(X_i)$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:

1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
2. Let $X = X_1 \dots X_n$ be uniformly random element of S
3. Then $\Pr[X_i = 1] \leq d/n$, so $H(X_i) \leq H(d/n)$
4. $\log |S| = H(X) \leq \sum_{i=1}^{n} H(X_i) \leq nH(d/n)$

Unexpected proofs: Information theory

- How much is $\sum_{i=0}^{d} \binom{n}{i}$, for $d \leq n/2$?
- At most $2^{nH(d/n)}$, where $H(\cdot)$ is binary entropy function
- Information-theoretic proof:
  1. Def $S = \{x \in \{0,1\}^n : |x| \leq d\}$, then $|S| = \sum_{i=0}^{d} \binom{n}{i}$
  2. Let $X = X_1 \dots X_n$ be uniformly random element of S
  3. Then $\Pr[X_i = 1] \leq d/n$, so $H(X_i) \leq H(d/n)$
  4. $\log |S| = H(X) \leq \sum_{i=1}^{n} H(X_i) \leq nH(d/n)$
  5. Exponentiating both sides finishes the proof

But that's just counting!

But that's just counting!

- Probabilistic arguments and information theory are just "counting arguments in disguise"

But that's just counting!

- Probabilistic arguments and information theory are just "counting arguments in disguise"
- That's true, but beside the point

But that's just counting!

- Probabilistic arguments and information theory are just "counting arguments in disguise"
- That's true, but beside the point
- The language of probability and information theory gives us intuitions and tools that wouldn't be readily available in the plain language of counting

But that's just counting!

- Probabilistic arguments and information theory are just "counting arguments in disguise"
- That's true, but beside the point
- The language of probability and information theory gives us intuitions and tools that wouldn't be readily available in the plain language of counting
- Large deviation inequalities, Lovász Local Lemma, chain rules, subadditivity of information,...

But that's just counting!

- Probabilistic arguments and information theory
  are just "counting arguments in disguise"
- That's true, but beside the point
- The language of probability and information theory gives us intuitions and tools that wouldn't be readily available in the plain language of counting
- Large deviation inequalities, Lovász Local Lemma,
  chain rules, subadditivity of information,. . .
- You *could* do those proofs in the language of counting,
  but you probably wouldn't find them

But that's just counting!

- Probabilistic arguments and information theory are just "counting arguments in disguise"
- That's true, but beside the point
- The language of probability and information theory gives us intuitions and tools that wouldn't be readily available in the plain language of counting
- Large deviation inequalities, Lovász Local Lemma, chain rules, subadditivity of information,. . .
- You *could* do those proofs in the language of counting, but you probably wouldn't find them
- Good to have probabilistic techniques in your tool-box

Unexpected proofs: Quantum

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.
- Why? Because quantum information is a rich melting pot of many branches of math: linear algebra, probability theory, group theory, geometry, combinatorics, functional analysis, . . .

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.
- Why? Because quantum information is a rich melting pot of many branches of math: linear algebra, probability theory, group theory, geometry, combinatorics, functional analysis, ...
- Bonus: no need to implement anything in the lab

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.
- Why? Because quantum information is a rich melting pot of many branches of math: linear algebra, probability theory, group theory, geometry, combinatorics, functional analysis, ...
- Bonus: no need to implement anything in the lab
- We'll give two examples:

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.
- Why? Because quantum information is a rich melting pot of many branches of math: linear algebra, probability theory, group theory, geometry, combinatorics, functional analysis, ...
- Bonus: no need to implement anything in the lab
- We'll give two examples:
  1. Lower bound on locally decodable codes [KW'03]

Unexpected proofs: Quantum

- We know quantum information & computation for its algorithms, crypto-schemes, communication protocols, non-locality, etc.
- This talk: using quantum techniques as a proof tool for things in *classical* CS, mathematics, etc.
- Why? Because quantum information is a rich melting pot of many branches of math: linear algebra, probability theory, group theory, geometry, combinatorics, functional analysis, ...
- Bonus: no need to implement anything in the lab
- We'll give two examples:
  1. Lower bound on locally decodable codes [KW'03]
  2. Lower bounds for linear programs [FMPTW'12]

But that's just linear algebra!

But that's just linear algebra!

- Quantum arguments are just "linear algebra in disguise"

But that's just linear algebra!

- Quantum arguments are just "linear algebra in disguise"
- That's true, but beside the point

But that's just linear algebra!

- Quantum arguments are just "linear algebra in disguise"
- That's true, but beside the point
- The language of quantum information and quantum algorithms gives us intuitions and tools that wouldn't be readily available in the plain language of linear algebra

But that's just linear algebra!

- Quantum arguments are just "linear algebra in disguise"
- That's true, but beside the point
- The language of quantum information and quantum algorithms gives us intuitions and tools that wouldn't be readily available in the plain language of linear algebra
- You *could* do those proofs in the language of linear algebra, but you probably wouldn't find them

But that's just linear algebra!

- Quantum arguments are just "linear algebra in disguise"
- That's true, but beside the point
- The language of quantum information and quantum algorithms gives us intuitions and tools that wouldn't be readily available in the plain language of linear algebra
- You *could* do those proofs in the language of linear algebra, but you probably wouldn't find them
- Good to have quantum techniques in your tool-box

Quantum computing reminder

Quantum computing reminder

- A state is a unit vector of complex amplitudes

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- *d*-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i |i\rangle \in \mathbb{C}^d$
- *n*-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i |i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$
- Measurement: specified by orthogonal projectors

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$
- Measurement: specified by orthogonal projectors
  $P_1, \ldots, P_k$, s.t. $\sum_{i=1}^{k} P_i = I$.

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$
- Measurement: specified by orthogonal projectors $P_1, \ldots, P_k$, s.t. $\sum_{i=1}^{k} P_i = I$.
  $\Pr[\text{outcome i}] = \mathrm{Tr}(P_i|\phi\rangle\langle\phi|)$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$
- Measurement: specified by orthogonal projectors $P_1, \ldots, P_k$, s.t. $\sum_{i=1}^{k} P_i = I$.
  Pr[outcome i] $=$ Tr$(P_i|\phi\rangle\langle\phi|)$
  State $|\phi\rangle$ then collapses to $P_i|\phi\rangle$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.
  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$
- Measurement: specified by orthogonal projectors $P_1, \ldots, P_k$, s.t. $\sum_{i=1}^{k} P_i = I$.
  Pr[outcome i] $= \mathrm{Tr}(P_i|\phi\rangle\langle\phi|)$
  State $|\phi\rangle$ then collapses to $P_i|\phi\rangle / \parallel P_i|\phi\rangle \parallel$

Quantum computing reminder

- A state is a unit vector of complex amplitudes
- Qubit: superposition $\alpha_0|0\rangle + \alpha_1|1\rangle \in \mathbb{C}^2$
- d-dimensional state: superposition $\sum_{i=1}^{d} \alpha_i|i\rangle \in \mathbb{C}^d$
- n-qubit state ($d = 2^n$): $|\phi\rangle = \sum_{i\in\{0,1\}^n} \alpha_i|i\rangle \in \mathbb{C}^{2^n}$

- Operations: unitary transform of the vector.

  Example: Hadamard gate $|b\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$

- Measurement: specified by orthogonal projectors $P_1, \ldots, P_k$, s.t. $\sum_{i=1}^{k} P_i = I$.

  Pr[outcome i] $= \mathrm{Tr}(P_i|\phi\rangle\langle\phi|)$

  State $|\phi\rangle$ then collapses to $P_i|\phi\rangle / \parallel P_i|\phi\rangle \parallel$

  Special case: $P_i = |i\rangle\langle i|$, then Pr[outcome i] $= |\alpha_i|^2$

Example 1:

Lower bounds for
locally decodable codes

Locally decodable codes

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

- Inefficient if you only want to decode a small part of x

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

- Inefficient if you only want to decode a small part of x

- C is k-query locally decodable if there is a decoder D that can decode individual bits x_i of x, while only looking at k bits of w

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

- Inefficient if you only want to decode a small part of x

- C is k-query locally decodable if there is a decoder D that can decode individual bits x_i of x, while only looking at k bits of w

- Hard question: optimal tradeoff between k and m?

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

- Inefficient if you only want to decode a small part of x

- C is k-query locally decodable if there is a decoder D that can decode individual bits x_i of x, while only looking at k bits of w

- Hard question: optimal tradeoff between k and m?

- Using quantum, we can show: $k = 2 \Rightarrow m \geq 2^{\Omega(n)}$

Locally decodable codes

- Error-correcting code: $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

  Decoder: if $w \in \{0,1\}^m$ is "close" to $C(x)$, then $D(w) = x$

- Inefficient if you only want to decode a small part of x
- C is k-query locally decodable if there is a decoder D that can decode individual bits x_i of x, while only looking at k bits of w
- Hard question: optimal tradeoff between k and m?
- Using quantum, we can show: $k = 2 \Rightarrow m \geq 2^{\Omega(n)}$
- Still the only superpolynomial bound known for LDCs

Example of 2-query LDC: Hadamard

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i}$$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x)$$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x) = e_i \cdot x$$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x) = e_i \cdot x = x_i$$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j\oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j\oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x) = e_i \cdot x = x_i$$

- With δm errors, $\Pr_j[w_j \neq C(x)_j] \leq \delta$

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x) = e_i \cdot x = x_i$$

- With δm errors, $\Pr_j[w_j \neq C(x)_j] \leq \delta$ and $\Pr_j[w_{j \oplus e_i} \neq C(x)_{j \oplus e_i}] \leq \delta$,

Example of 2-query LDC: Hadamard

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $C(x)$ is a codeword of 2^n bits
- Decoding x_i from corrupted codeword $w \approx C(x)$:
  1. pick random $j \in \{0,1\}^n$
  2. query w at positions j and $j \oplus e_i$
  3. output $w_j \oplus w_{j \oplus e_i}$
- This works perfectly if there is no noise ($w = C(x)$):

$$w_j \oplus w_{j \oplus e_i} = (j \cdot x) \oplus ((j \oplus e_i) \cdot x) = e_i \cdot x = x_i$$

- With δm errors, $\Pr_j[w_j \neq C(x)_j] \leq \delta$ and $\Pr_j[w_{j \oplus e_i} \neq C(x)_{j \oplus e_i}] \leq \delta$, so $\Pr[\text{we correctly output } x_i] \geq 1 - 2\delta$

Exponential lower bound [KW03]

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$:

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j, k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j,k) \in M_i$.

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j, k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j, k) \in M_i$.
  Measurement in basis $\{|j\rangle \pm |k\rangle\}$ gives $C(x)_j \oplus C(x)_k$.

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j,k) \in M_i$.
  Measurement in basis $\{|j\rangle \pm |k\rangle\}$ gives $C(x)_j \oplus C(x)_k$.
  But that's the output of the classical decoder, so equals x_i!

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j,k) \in M_i$.
  Measurement in basis $\{|j\rangle \pm |k\rangle\}$ gives $C(x)_j \oplus C(x)_k$.
  But that's the output of the classical decoder, so equals x_i!
- $|\phi_x\rangle$ has $\log m$ qubits, but predicts each of $x_1, \ldots, x_n$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}}\sum_{j=1}^m (-1)^{C(x)_j}|j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j,k) \in M_i$.
  Measurement in basis $\{|j\rangle \pm |k\rangle\}$ gives $C(x)_j \oplus C(x)_k$.
  But that's the output of the classical decoder, so equals x_i!
- $|\phi_x\rangle$ has $\log m$ qubits, but predicts each of $x_1, \dots, x_n$
- Random access code bound [Nayak'99]:
  $\log m \geq \Omega(n)$

Exponential lower bound [KW03]

- Given 2-query LDC $C : \{0,1\}^n \to \{0,1\}^m$.
  Normal form for the classical decoder of x_i [KT00]:
  query random (j,k) in matching M_i, output $C(x)_j \oplus C(x)_k$
- Def superposition over $C(x)$: $|\phi_x\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$
- We can predict x_i from $|\phi_x\rangle$: view M_i as a measurement with $m/2$ 2-dimensional projectors, $P_{jk} = |j\rangle\langle j| + |k\rangle\langle k|$
- Applying M_i to $|\phi_x\rangle$ gives
  $\frac{1}{\sqrt{2}}((-1)^{C(x)_j}|j\rangle + (-1)^{C(x)_k}|k\rangle)$ for random $(j,k) \in M_i$.
  Measurement in basis $\{|j\rangle \pm |k\rangle\}$ gives $C(x)_j \oplus C(x)_k$.
  But that's the output of the classical decoder, so equals x_i!
- $|\phi_x\rangle$ has $\log m$ qubits, but predicts each of $x_1, \ldots, x_n$
- Random access code bound [Nayak'99]:
  $\log m \geq \Omega(n) \Rightarrow m \geq 2^{\Omega(n)}$

Example 2:

Lower bounds for linear programs

Background: solving NP by linear programs?

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete *n*-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities
- Extended formulation: linear inequalities on $\binom{n}{2} + k$ variables s.t. projection on first $\binom{n}{2}$ variables gives TSP polytope

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities
- Extended formulation: linear inequalities on $\binom{n}{2} + k$ variables s.t. projection on first $\binom{n}{2}$ variables gives TSP polytope
- Swart'86 claimed polynomial-size extended formulation, which would give poynomial-time LP-algorithm for TSP

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities
- Extended formulation: linear inequalities on $\binom{n}{2} + k$ variables s.t. projection on first $\binom{n}{2}$ variables gives TSP polytope
- Swart'86 claimed polynomial-size extended formulation, which would give poynomial-time LP-algorithm for TSP
- Yannakakis'88: symmetric EFs for TSP are exponentially big

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities
- Extended formulation: linear inequalities on $\binom{n}{2} + k$ variables s.t. projection on first $\binom{n}{2}$ variables gives TSP polytope
- Swart'86 claimed polynomial-size extended formulation, which would give poynomial-time LP-algorithm for TSP
- Yannakakis'88: symmetric EFs for TSP are exponentially big
- Swart's LPs were symmetric, so they couldn't work

Background: solving NP by linear programs?

- Famous **P**-problem: linear programming [Khachian'79]
- Famous **NP**-hard problem: Traveling Salesman Problem
- TSP polytope: convex hull of all Hamiltonian cycles on complete n-vertex graph. This is a polytope in $\mathbb{R}^{\binom{n}{2}}$.
  TSP: minimize linear function over this polytope
  Unfortunately, polytope needs exponentially many inequalities
- Extended formulation: linear inequalities on $\binom{n}{2} + k$ variables s.t. projection on first $\binom{n}{2}$ variables gives TSP polytope
- Swart'86 claimed polynomial-size extended formulation, which would give poynomial-time LP-algorithm for TSP
- Yannakakis'88: symmetric EFs for TSP are exponentially big
- Swart's LPs were symmetric, so they couldn't work
- FMPTW'12 show the same for all extended formulations

Quantum vs classical communication complexity

Quantum vs classical communication complexity

- Communication complexity: Alice gets input $a \in \{0,1\}^k$, Bob gets input $b \in \{0,1\}^k$, they need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ with minimal communication

Quantum vs classical communication complexity

- Communication complexity: Alice gets input $a \in \{0,1\}^k$, Bob gets input $b \in \{0,1\}^k$, they need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ with minimal communication
- Nondeterministic communication complexity: protocol outputs 1 with positive probability on input a, b iff $f(a, b) = 1$

Quantum vs classical communication complexity

- Communication complexity: Alice gets input $a \in \{0,1\}^k$, Bob gets input $b \in \{0,1\}^k$, they need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ with minimal communication
- Nondeterministic communication complexity: protocol outputs 1 with positive probability on input a, b iff $f(a,b) = 1$
- W'00: exponential separation between quantum and classical nondeterministic protocols for support of the following $2^k \times 2^k$ matrix: $M_{ab} = (1 - a^T b)^2$

Quantum vs classical communication complexity

- Communication complexity: Alice gets input $a \in \{0,1\}^k$, Bob gets input $b \in \{0,1\}^k$, they need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ with minimal communication
- Nondeterministic communication complexity: protocol outputs 1 with positive probability on input a, b iff $f(a,b) = 1$
- W'00: exponential separation between quantum and classical nondeterministic protocols for support of the following $2^k \times 2^k$ matrix: $M_{ab} = (1 - a^T b)^2$
- Classical protocols need $\Omega(k)$ bits of communication for this

Quantum vs classical communication complexity

- Communication complexity: Alice gets input $a \in \{0,1\}^k$, Bob gets input $b \in \{0,1\}^k$, they need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$ with minimal communication
- Nondeterministic communication complexity: protocol outputs 1 with positive probability on input a, b iff $f(a, b) = 1$
- W'00: exponential separation between quantum and classical nondeterministic protocols for support of the following $2^k \times 2^k$ matrix: $M_{ab} = (1 - a^T b)^2$
- Classical protocols need $\Omega(k)$ bits of communication for this
- $\exists$ protocol for this using $O(\log k)$ qubits of communication

Lower bound for correlation polytope

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

Lower bound for correlation polytope

- Correlation polytope: $\text{COR}(k) = \text{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \text{COR}(k): \ \text{Tr}\left[(2\text{diag}(a) - aa^T)x\right] \leq 1$$

Slack of this constraint w.r.t. vertex $bb^T \in \text{COR}(k)$:

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
$S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right]$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
$S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
$S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2 = M_{ab}$

Lower bound for correlation polytope

- Correlation polytope: $\text{COR}(k) = \text{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \text{COR}(k): \ \text{Tr}\left[(2\text{diag}(a) - aa^T)x\right] \leq 1$$

  Slack of this constraint w.r.t. vertex $bb^T \in \text{COR}(k)$:
  $S_{ab} = 1 - \text{Tr}\left[(2\text{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2 = M_{ab}$
- Take slack matrix S for COR,
  with 2^k vertices bb^T for columns,
  2^k a-constraints for first 2^k rows,
  remaining inequalities for other rows

$$S = \left[\begin{array}{ccc} & \vdots & \\ \cdots & M_{ab} & \cdots \\ & \vdots & \\ \hline & \vdots & \end{array}\right]$$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

  Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
  $S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2 = M_{ab}$

- Take slack matrix S for COR,
  with 2^k vertices bb^T for columns,
  2^k a-constraints for first 2^k rows,
  remaining inequalities for other rows

$$S = \left[\begin{array}{ccc} & \vdots & \\ \cdots & M_{ab} & \cdots \\ & \vdots & \\ \hline & \vdots & \end{array}\right]$$

- $xc(\mathrm{COR}(k))$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

  Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
  $S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2 = M_{ab}$
- Take slack matrix S for COR,
  with 2^k vertices bb^T for columns,
  2^k a-constraints for first 2^k rows,
  remaining inequalities for other rows

$$S = \left[\begin{array}{ccc} & \vdots & \\ \cdots & M_{ab} & \cdots \\ & \vdots & \\ \hline & \vdots & \end{array}\right]$$

- $xc(\mathrm{COR}(k)) \overset{\text{Yannakakis}}{\geq} \exp(\text{nondetermin c.c. of } S)$

Lower bound for correlation polytope

- Correlation polytope: $\mathrm{COR}(k) = \mathrm{conv}\{bb^T \mid b \in \{0,1\}^k\}$
- For each $a \in \{0,1\}^k$, the following constraint hold:

$$\forall x \in \mathrm{COR}(k): \ \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)x\right] \leq 1$$

  Slack of this constraint w.r.t. vertex $bb^T \in \mathrm{COR}(k)$:
  $S_{ab} = 1 - \mathrm{Tr}\left[(2\mathrm{diag}(a) - aa^T)bb^T\right] = (1 - a^T b)^2 = M_{ab}$
- Take slack matrix S for COR,
  with 2^k vertices bb^T for columns,
  2^k a-constraints for first 2^k rows,
  remaining inequalities for other rows

$$S = \left[\begin{array}{ccc} & \vdots & \\ \cdots & M_{ab} & \cdots \\ & \vdots & \\ \hline & \vdots & \end{array}\right]$$

- $xc(\mathrm{COR}(k)) \overset{\text{Yannakakis}}{\geq} \exp(\text{nondetermin c.c. of } S) \geq 2^{\Omega(k)}$

Consequences

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems
- This refutes all P = NP "proofs" à la Swart

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems
- This refutes all P = NP "proofs" à la Swart
- Did we really *need* quantum for this proof?

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems
- This refutes all P = NP "proofs" à la Swart
- Did we really *need* quantum for this proof?
- No, we just needed to find the right matrix M and a classical nondeterministic communication lower bound

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems
- This refutes all P = NP "proofs" à la Swart
- Did we really *need* quantum for this proof?
- No, we just needed to find the right matrix M and a classical nondeterministic communication lower bound
- But the reason we found the right M is the earlier result about quantum communication complexity

Consequences

- We just showed that linear programs for optimizing over the correlation polytope need to be exponentially large
- This implies exponential lower bounds for TSP and other polytopes for NP-hard problems
- This refutes all P = NP "proofs" à la Swart
- Did we really *need* quantum for this proof?
- No, we just needed to find the right matrix M and a classical nondeterministic communication lower bound
- But the reason we found the right M is the earlier result about quantum communication complexity
- Wittgenstein: throw away the ladder after you climbed it

From quantum algorithms to polynomials

From quantum algorithms to polynomials

- "Polynomial method":
  efficient quantum algorithms $\Rightarrow$ low-degree polynomials

From quantum algorithms to polynomials

- "Polynomial method":
  efficient quantum algorithms $\Rightarrow$ low-degree polynomials

- Usual application: lower bounds on degree
  $\Rightarrow$ lower bounds on quantum complexity

From quantum algorithms to polynomials

- "Polynomial method":
  efficient quantum algorithms $\Rightarrow$ low-degree polynomials
- Usual application: lower bounds on degree
  $\Rightarrow$ lower bounds on quantum complexity
- But you can also use this method as a tool to construct low-degree polynomials with nice properties

From quantum algorithms to polynomials

- "Polynomial method": efficient quantum algorithms $\Rightarrow$ low-degree polynomials
- Usual application: lower bounds on degree $\Rightarrow$ lower bounds on quantum complexity
- But you can also use this method as a tool to construct low-degree polynomials with nice properties.

  Examples:

  - minimal-degree polynomial approximations to functions $f : \{0, \ldots, n\} \to \mathbb{R}$ [W08]

From quantum algorithms to polynomials

- "Polynomial method":
  efficient quantum algorithms $\Rightarrow$ low-degree polynomials
- Usual application: lower bounds on degree
  $\Rightarrow$ lower bounds on quantum complexity
- But you can also use this method as a tool to construct low-degree polynomials with nice properties.

  Examples:

  - minimal-degree polynomial approximations to functions $f : \{0, \ldots, n\} \to \mathbb{R}$ [W08]
  - quantum proof of Jackson's theorem [DW11]

Other examples of quantum proofs

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data
- Classical lower bound methods inspired by quantum methods

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data
- Classical lower bound methods inspired by quantum methods
- Aaronson: quantum reproofs of classical complexity results

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data
- Classical lower bound methods inspired by quantum methods
- Aaronson: quantum reproofs of classical complexity results
  - PP is closed under intersection [uses postselection]

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data
- Classical lower bound methods inspired by quantum methods
- Aaronson: quantum reproofs of classical complexity results
  - PP is closed under intersection [uses postselection]
  - Permanent is #P-hard [uses linear optics]

Other examples of quantum proofs

- Other uses of quantum information, often based on quantum encodings of classical data
- Classical lower bound methods inspired by quantum methods
- Aaronson: quantum reproofs of classical complexity results
  - PP is closed under intersection [uses postselection]
  - Permanent is #P-hard [uses linear optics]
- Results in functional analysis, other areas of math

Summary & Outlook

Summary & Outlook

- Quantum proofs for classical theorems

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, . . .

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a “bag of tricks” than a fully-developed “quantum method”

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a “bag of tricks”
  than a fully-developed “quantum method”
  (but you could say the same about probabilistic method)

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a "bag of tricks"
  than a fully-developed "quantum method"
  (but you could say the same about probabilistic method)

- Where can we find more applications?

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a "bag of tricks"
  than a fully-developed "quantum method"
  (but you could say the same about probabilistic method)

- Where can we find more applications?
  - Low-degree polynomials

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a “bag of tricks”
  than a fully-developed “quantum method”
  (but you could say the same about probabilistic method)

- Where can we find more applications?
  - Low-degree polynomials
  - Encoding-based lower bounds

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a "bag of tricks"
  than a fully-developed "quantum method"
  (but you could say the same about probabilistic method)

- Where can we find more applications?
  - Low-degree polynomials
  - Encoding-based lower bounds
  - Places where linear algebra and combinatorics meet

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a "bag of tricks"
  than a fully-developed "quantum method"
  (but you could say the same about probabilistic method)

- Where can we find more applications?

  - Low-degree polynomials
  - Encoding-based lower bounds
  - Places where linear algebra and combinatorics meet
  - ...

Summary & Outlook

- Quantum proofs for classical theorems

  Lower bounds for LDCs, linear programs, ...

- Currently this is more like a "bag of tricks"
  than a fully-developed "quantum method"
  (but you could say the same about probabilistic method)

- Where can we find more applications?

  - Low-degree polynomials
  - Encoding-based lower bounds
  - Places where linear algebra and combinatorics meet
  - ...

- Good to have quantum techniques in your tool-box!