Quantum algorithm for a generalized hidden shift problem

Andrew Childs
Caltech

Wim van Dam
UC Santa Barbara

What is the power of quantum computers?

Quantum mechanical computers can efficiently solve problems that classical computers (apparently) cannot.

- Manin/Feynman, early 1980s: Simulating quantum systems
- Deutsch 1985, Deutsch-Jozsa 1992, Bernstein-Vazirani 1993, Simon 1994: Black box problems
- Shor 1994: Factoring, discrete logarithm
- Many authors, late 1990s–Present: Some nonabelian hidden subgroup problems
- Freedman-Kitaev-Larsen 2000: Approximating Jones polynomial
- Hallgren 2002: Pell's equation
- van Dam-Hallgren-Ip 2002: Some hidden shift problems (e.g., shifted Legendre symbol)
- van Dam-Seroussi 2002: Estimating Gauss/Jacobi sums
- Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 2003: Black box graph traversal
- van Dam 2004, Kedlaya 2004: Approximately counting solutions of polynomial equations
- Hallgren 2005, Schmidt-Vollmer 2005: Finding unit/class groups of number fields

What is the power of quantum computers?

Quantum mechanical computers can efficiently solve problems that classical computers (apparently) cannot.

- Manin/Feynman, early 1980s: Simulating quantum systems
- Deutsch 1985, Deutsch-Jozsa 1992, Bernstein-Vazirani 1993, Simon 1994: Black box problems
- Shor 1994: Factoring, discrete logarithm
- Many authors, late 1990s–Present: Some nonabelian hidden subgroup problems
- Freedman-Kitaev-Larsen 2000: Approximating Jones polynomial
- Hallgren 2002: Pell's equation
- van Dam-Hallgren-Ip 2002: Some hidden shift problems (e.g., shifted Legendre symbol)
- van Dam-Seroussi 2002: Estimating Gauss/Jacobi sums
- Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 2003: Black box graph traversal
- van Dam 2004, Kedlaya 2004: Approximately counting solutions of polynomial equations
- Hallgren 2005, Schmidt-Vollmer 2005: Finding unit/class groups of number fields

Questions:

What is the power of quantum computers?

Quantum mechanical computers can efficiently solve problems that classical computers (apparently) cannot.

- Manin/Feynman, early 1980s: Simulating quantum systems
- Deutsch 1985, Deutsch-Jozsa 1992, Bernstein-Vazirani 1993, Simon 1994: Black box problems
- Shor 1994: Factoring, discrete logarithm
- Many authors, late 1990s–Present: Some nonabelian hidden subgroup problems
- Freedman-Kitaev-Larsen 2000: Approximating Jones polynomial
- Hallgren 2002: Pell's equation
- van Dam-Hallgren-Ip 2002: Some hidden shift problems (e.g., shifted Legendre symbol)
- van Dam-Seroussi 2002: Estimating Gauss/Jacobi sums
- Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 2003: Black box graph traversal
- van Dam 2004, Kedlaya 2004: Approximately counting solutions of polynomial equations
- Hallgren 2005, Schmidt-Vollmer 2005: Finding unit/class groups of number fields

Questions:

- What is the computational power of quantum mechanics?

What is the power of quantum computers?

Quantum mechanical computers can efficiently solve problems that classical computers (apparently) cannot.

- Manin/Feynman, early 1980s: Simulating quantum systems
- Deutsch 1985, Deutsch-Jozsa 1992, Bernstein-Vazirani 1993, Simon 1994: Black box problems
- Shor 1994: Factoring, discrete logarithm
- Many authors, late 1990s–Present: Some nonabelian hidden subgroup problems
- Freedman-Kitaev-Larsen 2000: Approximating Jones polynomial
- Hallgren 2002: Pell's equation
- van Dam-Hallgren-Ip 2002: Some hidden shift problems (e.g., shifted Legendre symbol)
- van Dam-Seroussi 2002: Estimating Gauss/Jacobi sums
- Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 2003: Black box graph traversal
- van Dam 2004, Kedlaya 2004: Approximately counting solutions of polynomial equations
- Hallgren 2005, Schmidt-Vollmer 2005: Finding unit/class groups of number fields

Questions:

- What is the computational power of quantum mechanics?
- Is public-key cryptography possible in a quantum world?
  Shor's algorithm breaks RSA, elliptic curve cryptosystems, Diffie-Hellman key exchange, etc.
  What about, e.g., lattice cryptosystems?

Generalized hidden shift problem

Given: $f(b,x) : \{0,1,\ldots,M-1\} \times \mathbb{Z}_N \to S$

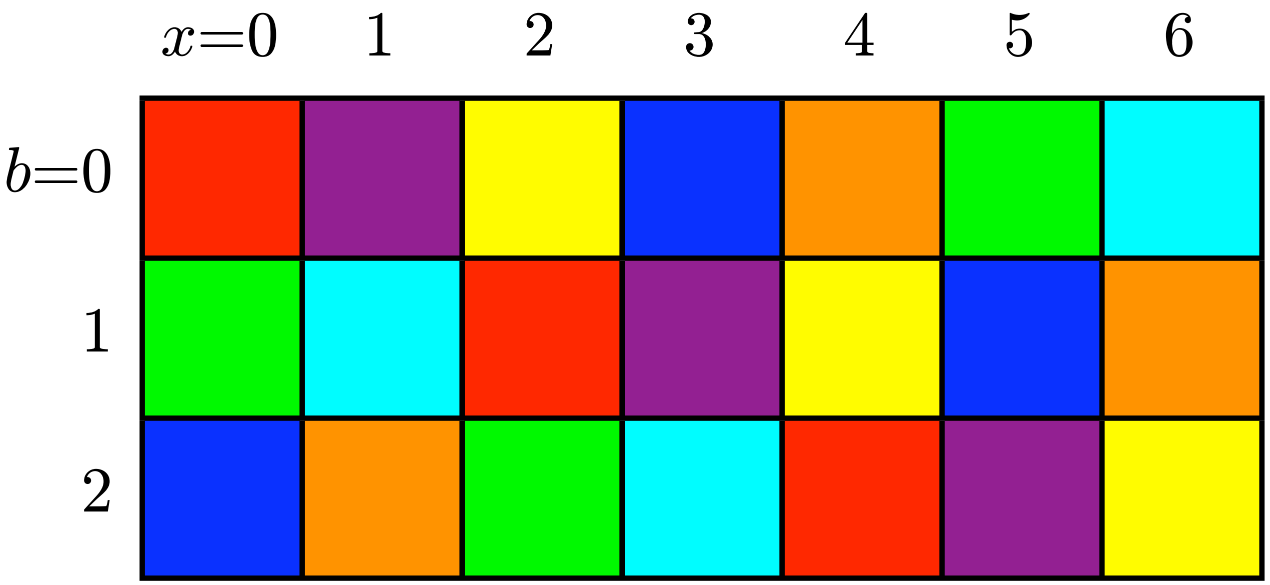
Satisfying: $f(0,x)$ injective

$$f(b+1, x+s) = f(b,x)$$

Find: s (the *hidden shift*)

$M = 2$ (hardest), … , N (easiest)

Example. $N = 7, M = 3, s = 2$



Classical complexity

Claim. To determine s, a classical, randomized algorithm must make exponentially many queries (in $\log N$) to f.

Classical complexity

Claim. To determine s, a classical, randomized algorithm must make exponentially many queries (in $\log N$) to f.

Proof idea:

- Since the function values are arbitrary, they are not informative until we find two inputs that give the same output.
- The probability of seeing such a collision is very small unless $\# \text{ queries} \gtrsim \sqrt{N}$ (birthday problem). Hence $\Omega(\sqrt{N})$ queries are needed.

Classical complexity

Claim. To determine s, a classical, randomized algorithm must make exponentially many queries (in $\log N$) to f.

Proof idea:

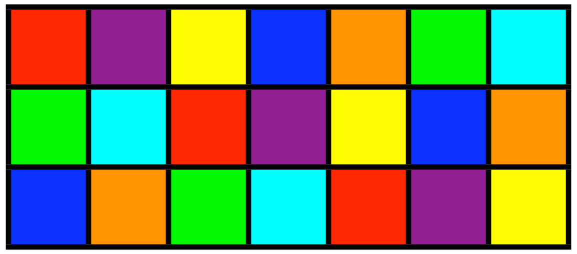
- Since the function values are arbitrary, they are not informative until we find two inputs that give the same output.
- The probability of seeing such a collision is very small unless $\#\text{ queries} \gtrsim \sqrt{N}$ (birthday problem). Hence $\Omega(\sqrt{N})$ queries are needed.

Note: This holds independent of how big M is.

Quantum query complexity

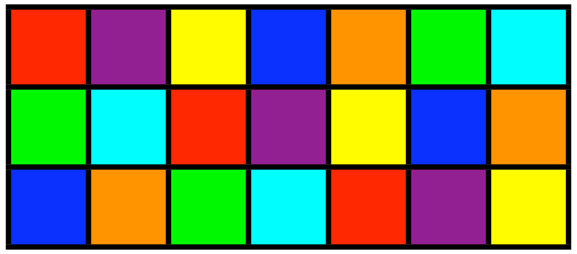
Quantum query complexity

Query f in superposition:

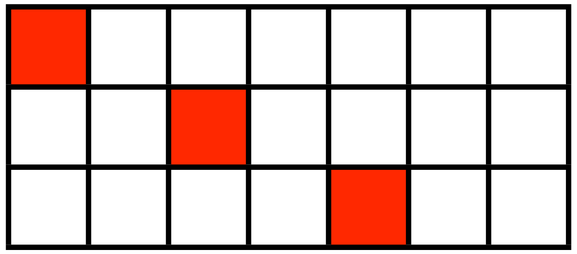


Quantum query complexity

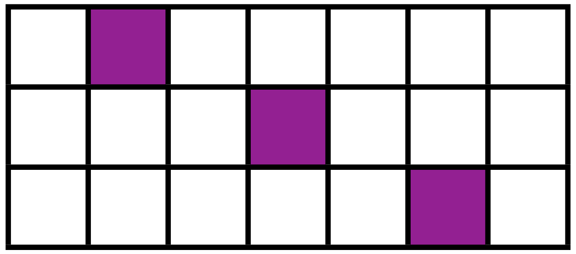
Query f in superposition:



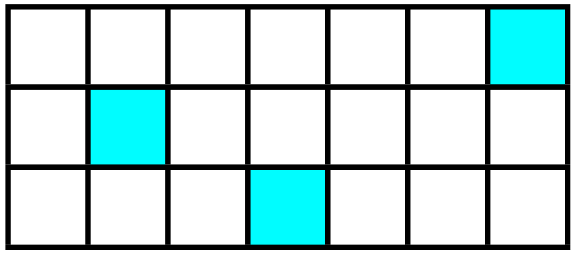
Measure function value: obtain (with equal probability)



or

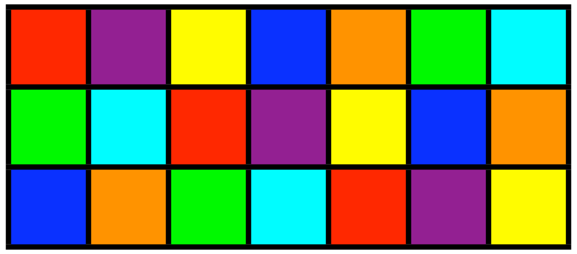


or $\cdots$ or

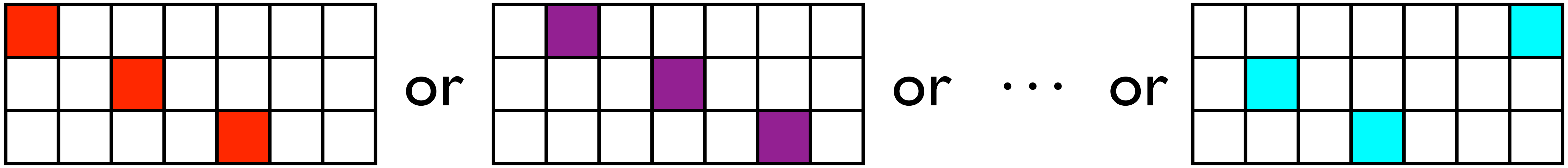


Quantum query complexity

Query f in superposition:



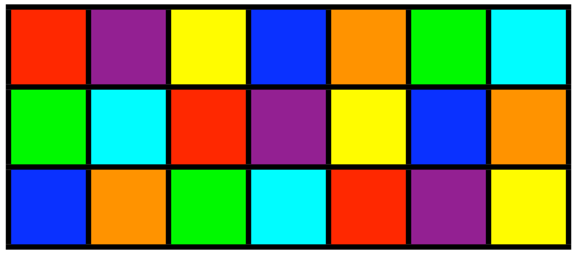
Measure function value: obtain (with equal probability)



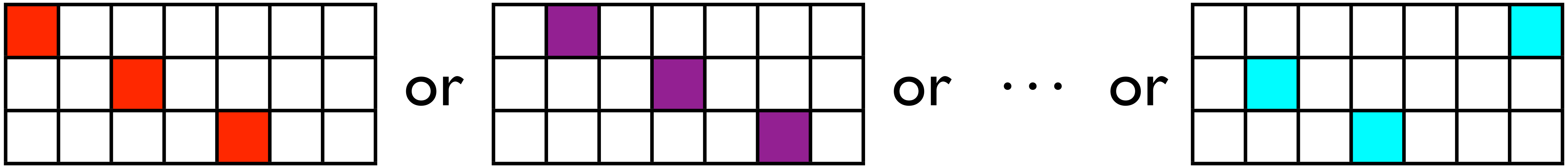
The quantum states for different values of s are far apart, so they can be distinguished using only a few copies ($k \leq \mathrm{poly}(\log N)$, again independent of M).

Quantum query complexity

Query f in superposition:



Measure function value: obtain (with equal probability)

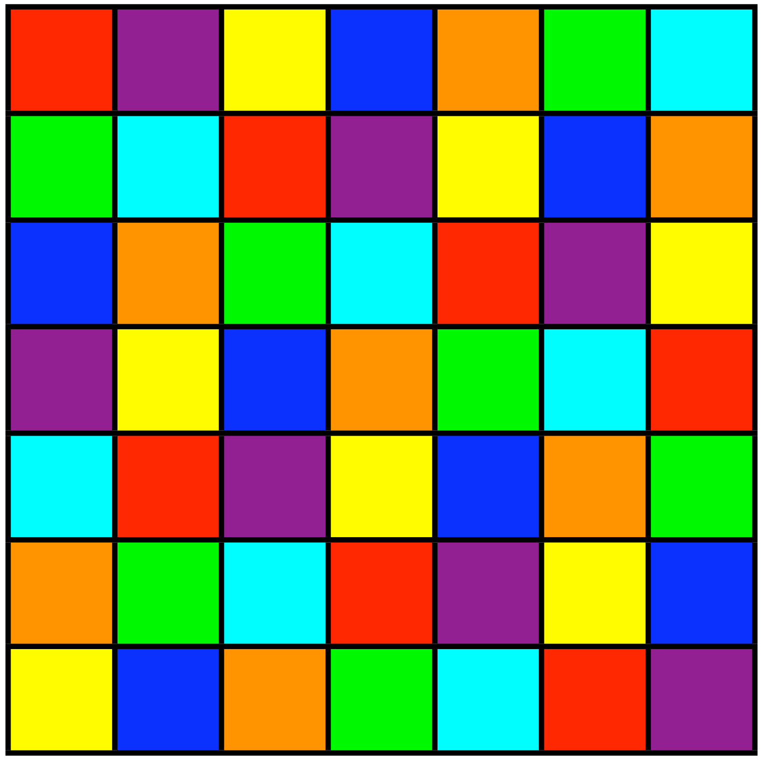


The quantum states for different values of s are far apart, so they can be distinguished using only a few copies ($k \leq \mathrm{poly}(\log N)$, again independent of M).

Main question: *Can we do it in* $\mathrm{poly}(\log N)$ *time?*

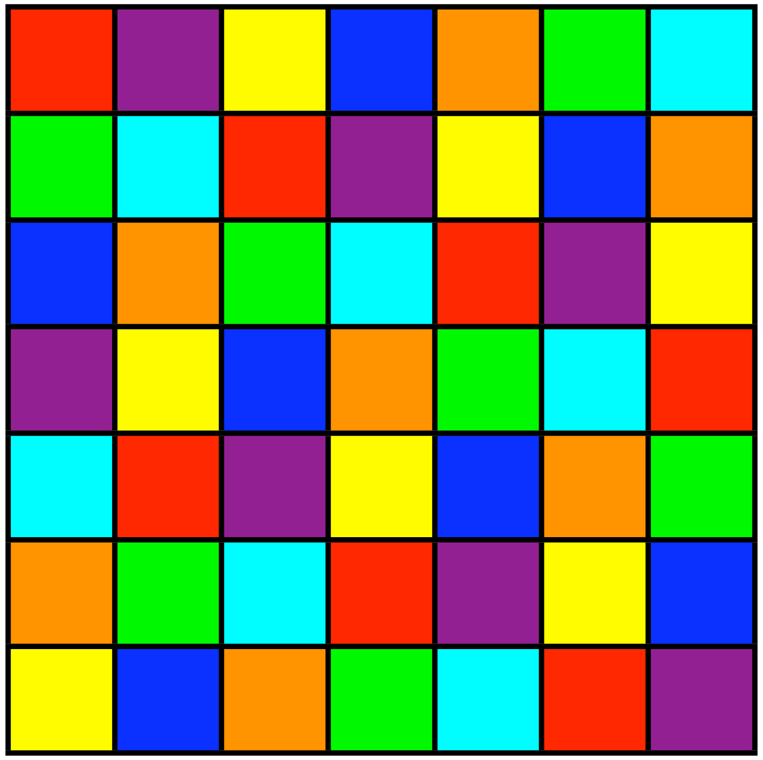
$M=N$: An abelian hidden subgroup problem

Easiest hidden shift problem:



$M{=}N$: An abelian hidden subgroup problem

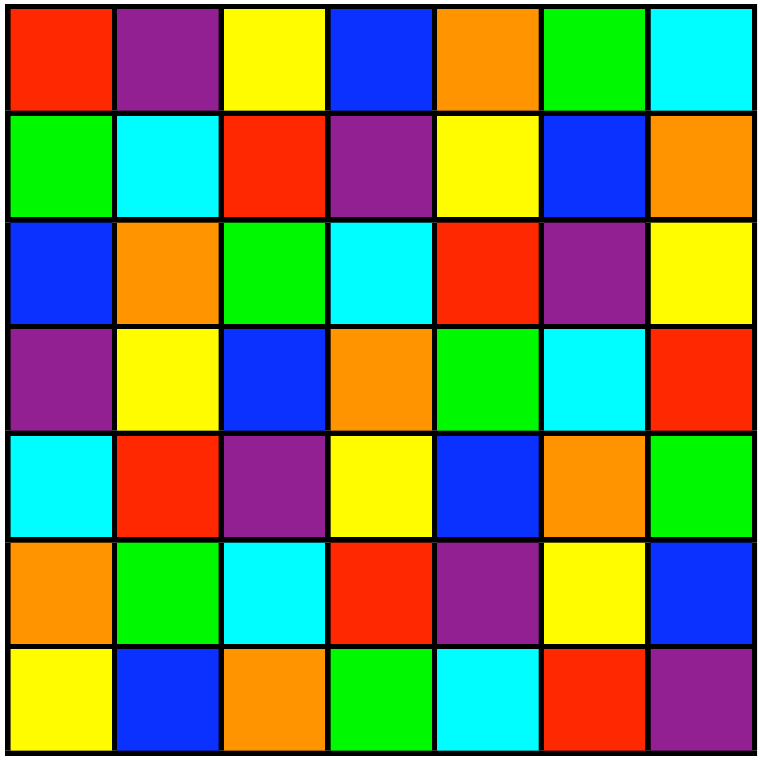
Easiest hidden shift problem:



This is an instance of the *hidden subgroup problem* in the abelian group $G = \mathbb{Z}_N \times \mathbb{Z}_N$. Shor's algorithm ("Fourier transform and measure") finds s efficiently.

$M = N$: An abelian hidden subgroup problem

Easiest hidden shift problem:

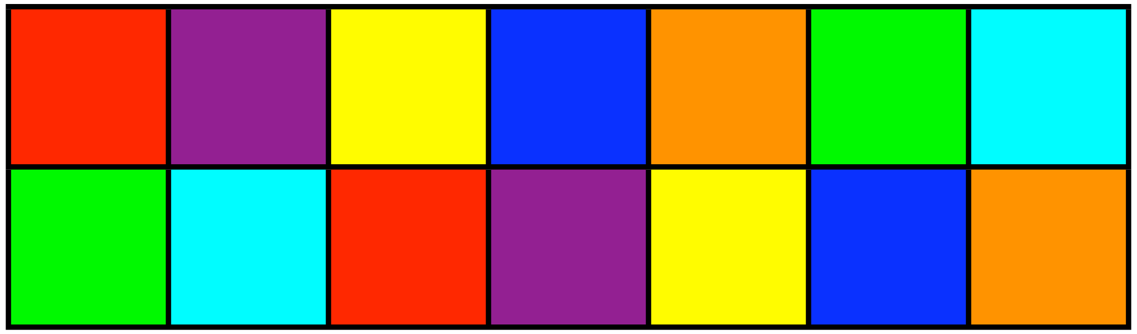


This is an instance of the *hidden subgroup problem* in the abelian group $G = \mathbb{Z}_N \times \mathbb{Z}_N$. Shor's algorithm ("Fourier transform and measure") finds s efficiently.

The same approach works for any $M \geq N/\mathrm{poly}(\log N)$, but not smaller!

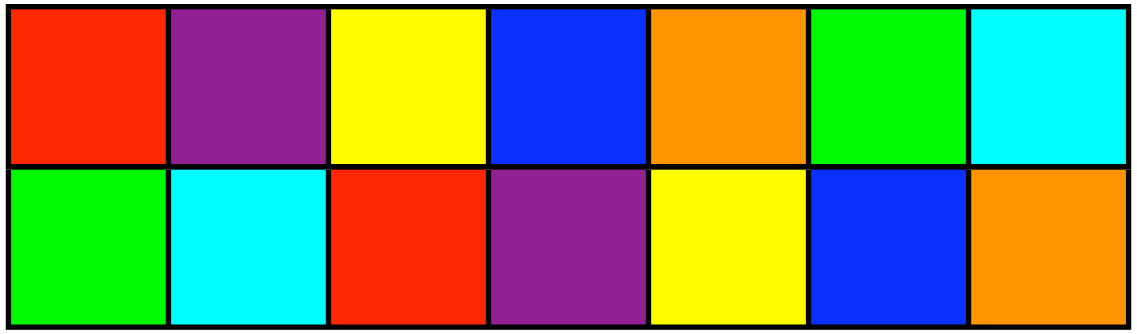
$M{=}2$: The dihedral hidden subgroup problem

Hardest hidden shift problem:



$M{=}2$: The dihedral hidden subgroup problem

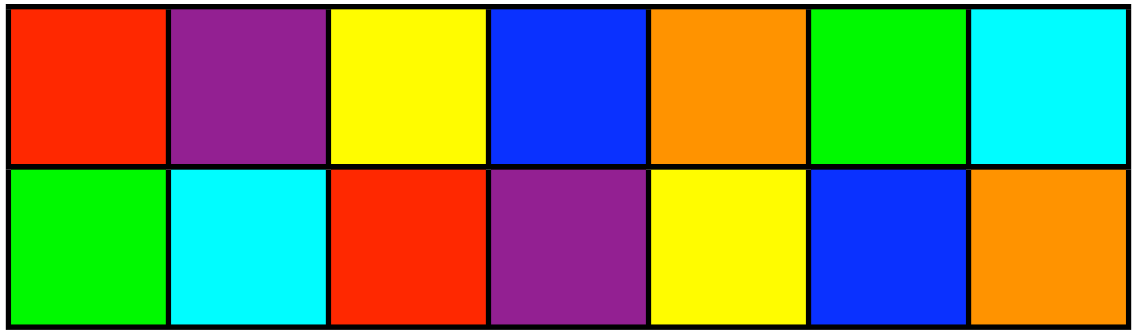
Hardest hidden shift problem:



This is also a hidden subgroup problem, but now in a nonabelian group, the *dihedral group* $G = \mathbb{Z}_2 \ltimes \mathbb{Z}_N$.

$M{=}2$: The dihedral hidden subgroup problem

Hardest hidden shift problem:

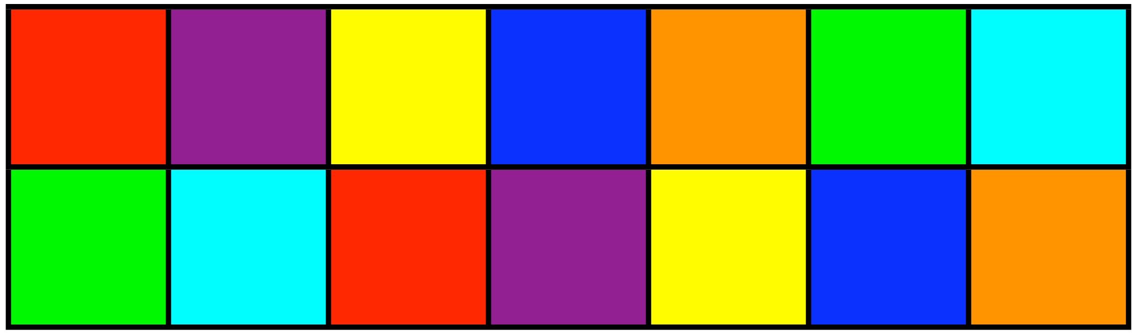


This is also a hidden subgroup problem, but now in a nonabelian group, the *dihedral group* $G = \mathbb{Z}_2 \ltimes \mathbb{Z}_N$.

Regev 2002: Solution to the DHSP can be used to find short vectors in lattices ($\sqrt{n}$-unique-SVP), which would break, e.g., the Ajtai-Dwork cryptosystem.

$M{=}2$: The dihedral hidden subgroup problem

Hardest hidden shift problem:



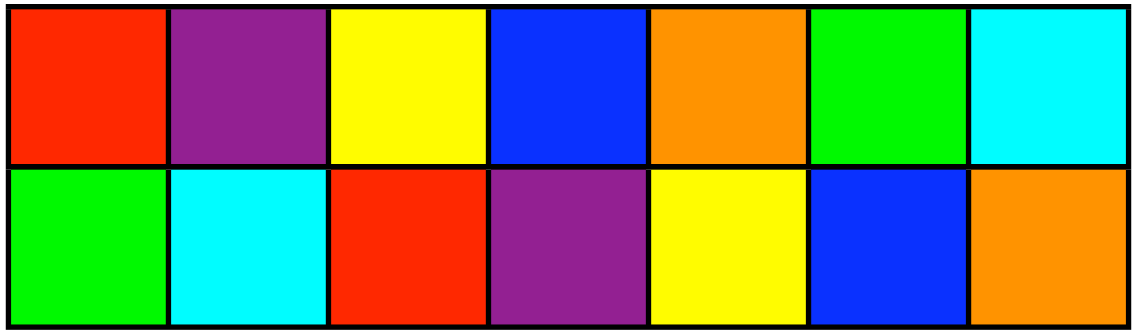
This is also a hidden subgroup problem, but now in a nonabelian group, the *dihedral group* $G = \mathbb{Z}_2 \ltimes \mathbb{Z}_N$.

Regev 2002: Solution to the DHSP can be used to find short vectors in lattices ($\sqrt{n}$-unique-SVP), which would break, e.g., the Ajtai-Dwork cryptosystem.

Kuperberg 2003: Algorithm with run time $2^{O(\sqrt{\log N})}$.

$M{=}2$: The dihedral hidden subgroup problem

Hardest hidden shift problem:



This is also a hidden subgroup problem, but now in a nonabelian group, the *dihedral group* $G = \mathbb{Z}_2 \ltimes \mathbb{Z}_N$.

Regev 2002: Solution to the DHSP can be used to find short vectors in lattices ($\sqrt{n}$-unique-SVP), which would break, e.g., the Ajtai-Dwork cryptosystem.

Kuperberg 2003: Algorithm with run time $2^{O(\sqrt{\log N})}$.

Regev's reduction also works for larger M. Is this any easier?

Main result

Theorem. Let $M = N^\epsilon$ for any fixed $\epsilon > 0$. Then there is an efficient (i.e., run time $\mathrm{poly}(\log N)$) quantum algorithm for the generalized hidden shift problem, using entangled measurements on $k = \max\{3, \log \frac{1}{\epsilon}\}$ registers.

Main result

Theorem. Let $M = N^\epsilon$ for any fixed $\epsilon > 0$. Then there is an efficient (i.e., run time $\mathrm{poly}(\log N)$) quantum algorithm for the generalized hidden shift problem, using entangled measurements on $k = \max\{3, \log \frac{1}{\epsilon}\}$ registers.

Note: Unfortunately, this is not good enough to get better-than-classical algorithms for lattice problems. (That seems to require $M = \mathrm{poly}(\log N)$.)

Main result

Theorem. Let $M = N^\epsilon$ for any fixed $\epsilon > 0$. Then there is an efficient (i.e., run time $\mathrm{poly}(\log N)$) quantum algorithm for the generalized hidden shift problem, using entangled measurements on $k = \max\{3, \log \frac{1}{\epsilon}\}$ registers.

Note: Unfortunately, this is not good enough to get better-than-classical algorithms for lattice problems. (That seems to require $M = \mathrm{poly}(\log N)$.)

Tools:

- "Pretty good measurement" on hidden shift states, à la Bacon, Childs, van Dam 2005.
- Integer programming in constant dimensions (Lenstra 1983).

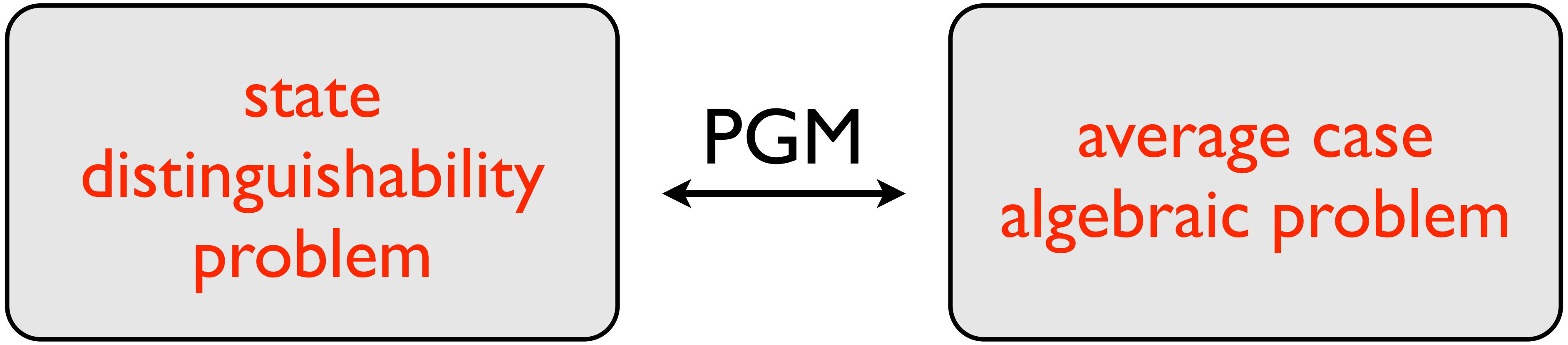
Pretty good measurement

PGM: A particularly nice, and often optimal, measurement for distinguishing members of an ensemble of quantum states.

Pretty good measurement

PGM: A particularly nice, and often optimal, measurement for distinguishing members of an ensemble of quantum states.

For certain semidirect HSPs (BCD 05) and hidden shift problems (this talk):



state distinguishability problem

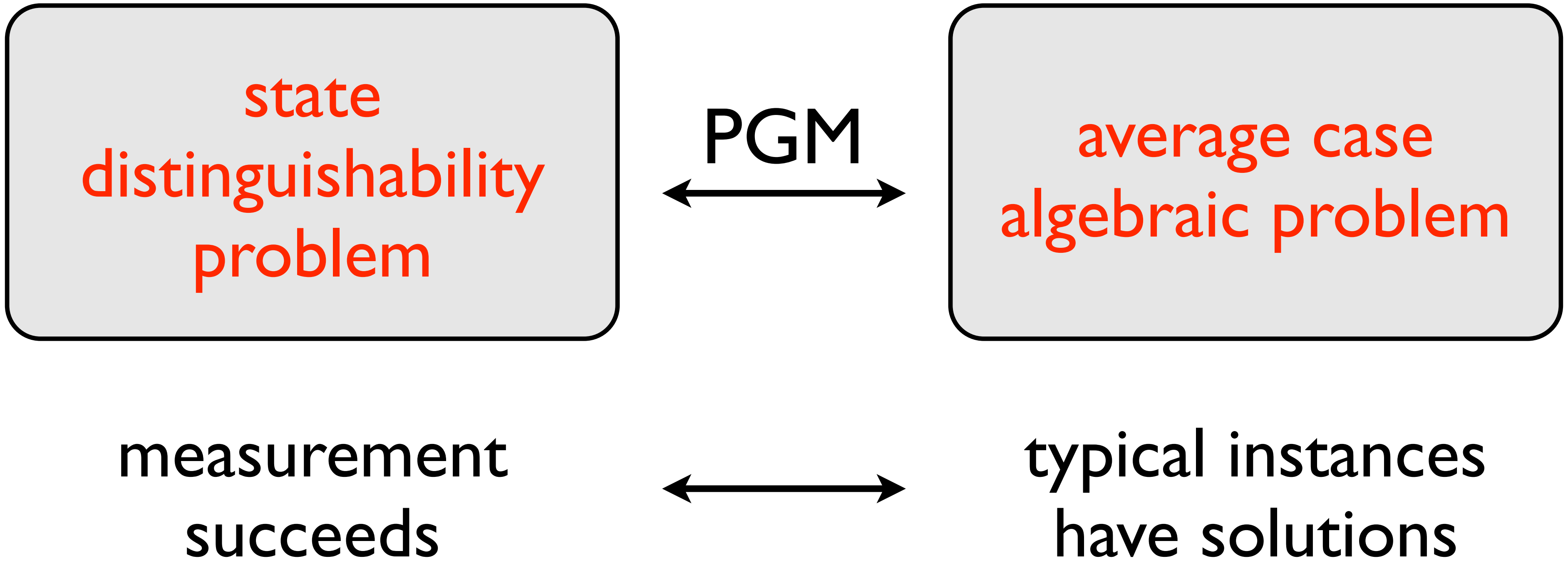
PGM

average case algebraic problem

Pretty good measurement

PGM: A particularly nice, and often optimal, measurement for distinguishing members of an ensemble of quantum states.

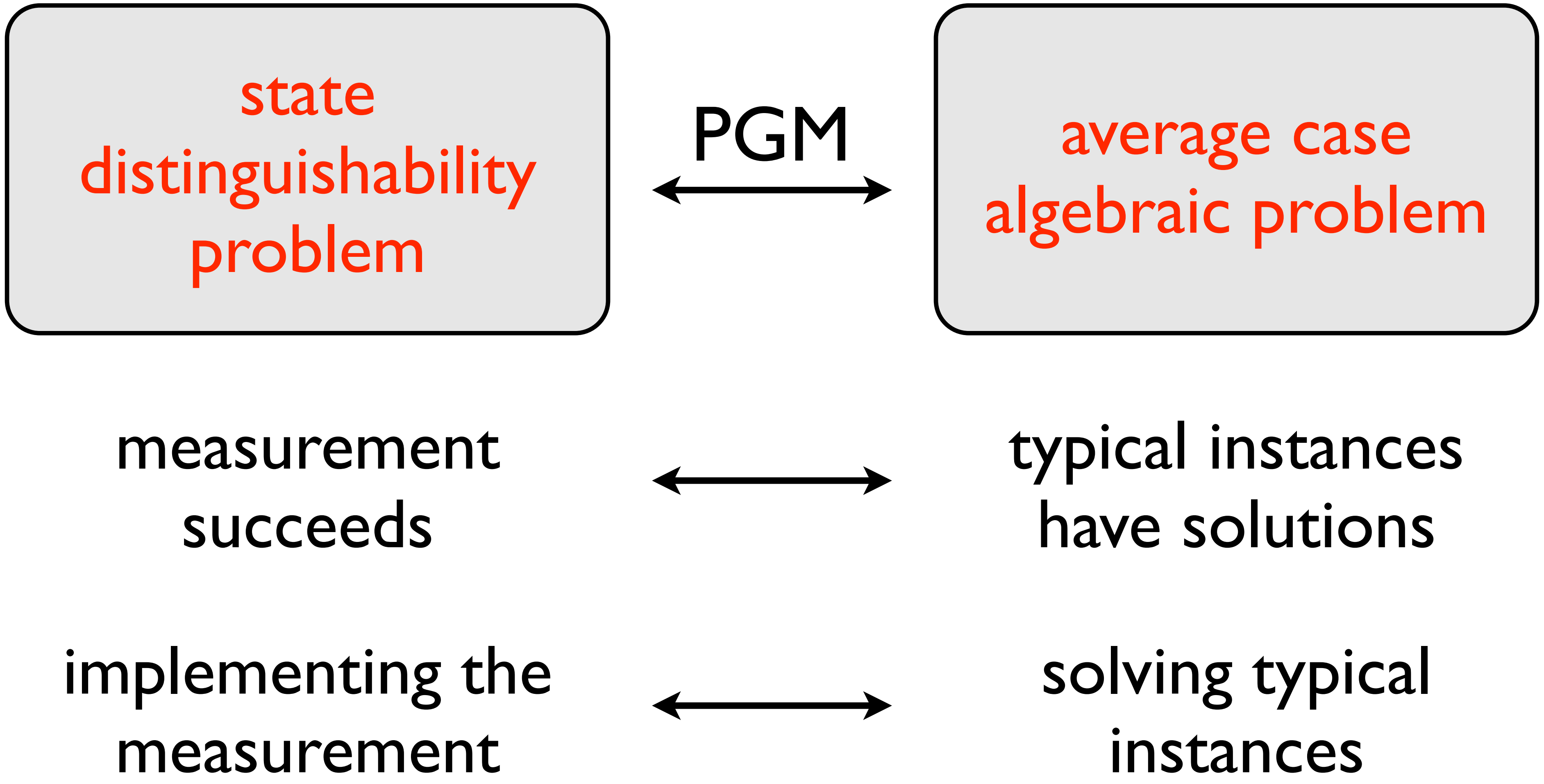
For certain semidirect HSPs (BCD 05) and hidden shift problems (this talk):



Pretty good measurement

PGM: A particularly nice, and often optimal, measurement for distinguishing members of an ensemble of quantum states.

For certain semidirect HSPs (BCD 05) and hidden shift problems (this talk):



state distinguishability problem

PGM

average case algebraic problem

measurement succeeds

typical instances have solutions

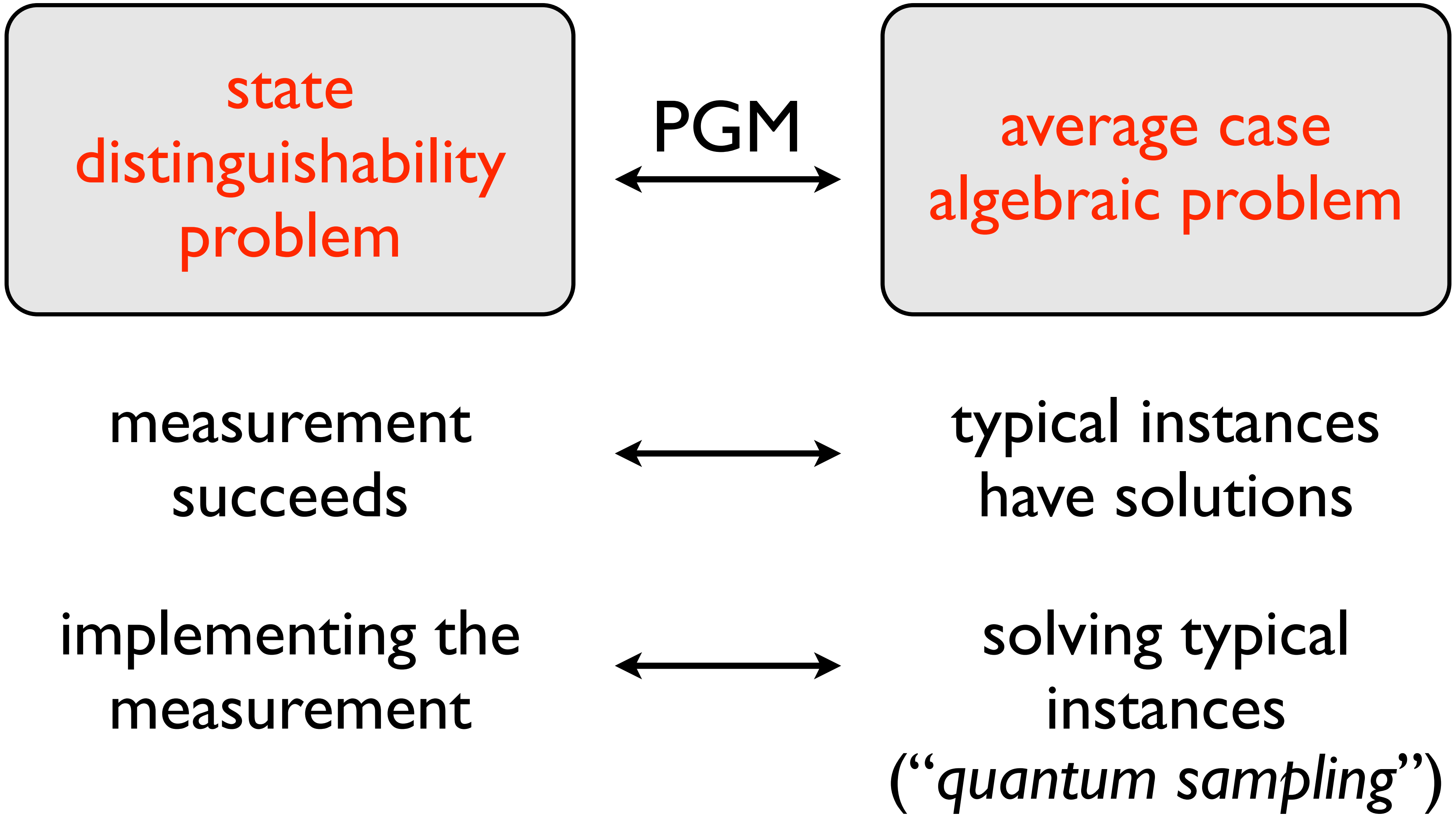
implementing the measurement

solving typical instances

Pretty good measurement

PGM: A particularly nice, and often optimal, measurement for distinguishing members of an ensemble of quantum states.

For certain semidirect HSPs (BCD 05) and hidden shift problems (this talk):



state distinguishability problem ⟷ (PGM) ⟷ average case algebraic problem

measurement succeeds ⟷ typical instances have solutions

implementing the measurement ⟷ solving typical instances (“*quantum sampling*”)

The algebraic problem

Given: random $x \in \mathbb{Z}_N^k$

random $w \in \mathbb{Z}_N$

Find: $b \in \{0, 1, \ldots, M-1\}^k$

such that $b \cdot x = w \bmod N$

The algebraic problem

Given: random $x \in \mathbb{Z}_N^k$

random $w \in \mathbb{Z}_N$

Find: $b \in \{0, 1, \ldots, M-1\}^k$

such that $b \cdot x = w \bmod N$

Key observation: This is a k-dimensional integer program.

- Solutions of $b \cdot x = w$ over $\mathbb{Z}$ form a shifted integer lattice
- "$\mathrm{mod}\ N$" can be enforced by adding a component
- $0 \leq b_j \leq M-1$ is a pair of linear constraints

The algebraic problem

Given: random $x \in \mathbb{Z}_N^k$

random $w \in \mathbb{Z}_N$

Find: $b \in \{0, 1, \ldots, M-1\}^k$

such that $b \cdot x = w \bmod N$

Key observation: This is a k-dimensional integer program.

- Solutions of $b \cdot x = w$ over $\mathbb{Z}$ form a shifted integer lattice
- "$\mathrm{mod}\ N$" can be enforced by adding a component
- $0 \leq b_j \leq M-1$ is a pair of linear constraints

Lenstra 1983: $2^{O(k^3)}$ time algorithm for integer programming in k dimensions (using LLL lattice basis reduction)

Analysis of typical number of solutions

$b \cdot x = w \bmod N$

Analysis of typical number of solutions

$$b \cdot x = w \bmod N$$

of b's

Expected number of solutions: $\dfrac{M^k}{N}$

# of values of w

Analysis of typical number of solutions

$$b \cdot x = w \bmod N$$

Expected number of solutions: $$\frac{M^k \longleftarrow \text{\# of } b\text{'s}}{N \longleftarrow \text{\# of values of } w}$$

So we expect to need $k \approx \dfrac{\log N}{\log M}$ copies. $(M = N^\epsilon \leftrightarrow k = \frac{1}{\epsilon})$

Analysis of typical number of solutions

$$b \cdot x = w \bmod N$$

Expected number of solutions: $$\frac{M^k \longleftarrow \text{\# of } b\text{'s}}{N \longleftarrow \text{\# of values of } w}$$

So we expect to need $k \approx \dfrac{\log N}{\log M}$ copies. $(M = N^\epsilon \leftrightarrow k = \frac{1}{\epsilon})$

Proof idea: Second moment method.

Analysis of typical number of solutions

$$b \cdot x = w \bmod N$$

Expected number of solutions: $$\frac{M^k \longleftarrow \text{\# of } b\text{'s}}{N \longleftarrow \text{\# of values of } w}$$

So we expect to need $k \approx \dfrac{\log N}{\log M}$ copies. $\left(M = N^{\epsilon} \leftrightarrow k = \frac{1}{\epsilon}\right)$

Proof idea: Second moment method.

Lemma. (used to bound variance)

For any fixed b, the number of solutions $x \in \mathbb{Z}_N^k$ to the equation $b \cdot x = 0 \bmod N$ is $N^{k-1}\gcd(b_1, \ldots, b_k, N)$.

Analysis of typical number of solutions

$$b \cdot x = w \bmod N$$

Expected number of solutions: $\dfrac{M^k}{N}$

of b's

of values of w

So we expect to need $k \approx \dfrac{\log N}{\log M}$ copies. $(M = N^\epsilon \leftrightarrow k = \frac{1}{\epsilon})$

Proof idea: Second moment method.

Lemma. (used to bound variance)

For any fixed b, the number of solutions $x \in \mathbb{Z}_N^k$ to the equation $b \cdot x = 0 \bmod N$ is $N^{k-1} \gcd(b_1, \ldots, b_k, N)$.

$\square$

Questions

- Is the quantum solvability of the generalized hidden shift problem with $M = \Omega(N^\epsilon)$ useful for any problems going beyond factoring/discrete log?

- Can we solve the problem efficiently for smaller M?
  Can we at least interpolate with Kuperberg's algorithm?

- What if we replace $\mathbb{Z}_N$ by a nonabelian group?
  (Then even $M=2$ is not a hidden subgroup problem.)
  Can we solve this even for very large M?