Quantum query complexity of symmetric problems.

Daniel Copeland and Jamie Pommersheim

JMM
San Diego

January 11, 2018

Guess the permutation!

Consider a group of permutations G acting on a finite set Ω.

Suppose someone draws an element g uniformly from G and hides it in a black box (or oracle).

We can access the hidden permutation by submitting an element $\alpha \in \Omega$ to the oracle and the oracle replies with the element $g \cdot \alpha$.

Guess the permutation!

Consider a group of permutations G acting on a finite set Ω.

Suppose someone draws an element g uniformly from G and hides it in a black box (or oracle).

We can access the hidden permutation by submitting an element $\alpha \in \Omega$ to the oracle and the oracle replies with the element $g \cdot \alpha$.

How many accesses (or queries) are needed to determine the unknown element g?

Guess the permutation!

Consider a group of permutations G acting on a finite set Ω.

Suppose someone draws an element g uniformly from G and hides it in a black box (or oracle).

We can access the hidden permutation by submitting an element $\alpha \in \Omega$ to the oracle and the oracle replies with the element $g \cdot \alpha$.

How many accesses (or queries) are needed to determine the unknown element g?

Query complexity.

The answer is a long-studied invariant of permutation groups called the **minimum base size** (or just **base size**), denoted $b(G)$.

For instance, $b(S_n) = n - 1$ and $b(A_n) = n - 2$.

The quantum version

To "quantize" this problem is essentially to linearize it. Every group element $g \in G$ corresponds to a unitary operator

$$\pi(g) : \mathbb{C}\Omega \to \mathbb{C}\Omega.$$

The quantum version

To "quantize" this problem is essentially to linearize it. Every group element $g \in G$ corresponds to a unitary operator

$$\pi(g) : \mathbb{C}\Omega \to \mathbb{C}\Omega.$$

(So each g acts by a permutation matrix in the natural basis).

The quantum version

To "quantize" this problem is essentially to linearize it. Every group element $g \in G$ corresponds to a unitary operator

$$\pi(g) : \mathbb{C}\Omega \to \mathbb{C}\Omega.$$

(So each g acts by a permutation matrix in the natural basis).

A quantum algorithm to determine the permutation g consists of choosing input states, feeding them to the hidden unitary $\pi(g)$, measuring the result, and making a guess based on the measurement.

Result

Theorem. (C., Pommersheim)

An optimal t-query quantum algorithm successfully identifies the permutation g with probability

$$P_{\text{success}} = \frac{1}{|G|} \sum_{\chi} dim(\chi)^2$$

where the sum is over all irreps which appear in $\mathbb{C}\Omega^{\otimes t}$.

Quantum query complexity

Corollary.

The **exact quantum query complexity** of Guess the Permutation! with G acting on Ω is

$$\gamma := \min_t \{\text{every irrep of } G \text{ appears in } \mathbb{C}\Omega^{\otimes t}\}.$$

Quantum query complexity

Corollary.

The **exact quantum query complexity** of Guess the Permutation! with G acting on Ω is

$$\gamma := \min_t\{\text{every irrep of } G \text{ appears in } \mathbb{C}\Omega^{\otimes t}\}.$$

The **bounded error quantum query complexity** of Guess the Permutation! is

$$\gamma^{\text{bdd}} := \min_t\{\sum_{\chi} \dim(\chi)^2 \geq 2/3\}$$

with the sum taken over all irreps χ appearing in $\mathbb{C}\Omega^{\otimes t}$.

Quantum query complexity

Corollary.

The **exact quantum query complexity** of Guess the Permutation! with G acting on Ω is

$$\gamma := \min_t\{\text{every irrep of } G \text{ appears in } \mathbb{C}\Omega^{\otimes t}\}.$$

The **bounded error quantum query complexity** of Guess the Permutation! is

$$\gamma^{\text{bdd}} := \min_t\{\sum_{\chi} \dim(\chi)^2 \geq 2/3\}$$

with the sum taken over all irreps χ appearing in $\mathbb{C}\Omega^{\otimes t}$.

$$\gamma^{\text{bdd}} \leq \gamma \leq b(G).$$

Strict inequalities mean quantum speedups.

Special case

The full symmetric group

$G = S_n$ (with $\Omega = \{1, \ldots, n\}$. This is the defining rep'n of S_n and splits as

$$\mathbb{C}\Omega = V_{[n]} + V_{[n-1,1]}.$$

Taking the t-th tensor power of this rep'n we get all irreps whose corresponding Young diagrams have $\geq n - t$ columns. Therefore:

$$\gamma = n - 1$$

(this many queries are needed to get the sign representation $[1^n]$) and

$$\gamma^{\mathrm{bdd}} = n - 2\sqrt{n} + O(n^{1/6})$$

(deduced using RSK correspondence, Baik-Deift-Johannson Theorem, Tracy-Widom distribution).

Computer example

Mathieu Group

Take $G = M_{12}$ which acts sharply 5-transitively on 12 points. Therefore $b(M_{12}) = 5$. Using GAP, we see that $\gamma = 4$ (ie every irrep of M_{12} appears in the 4th tensor power of the defining rep, and no sooner). So we get a 1-query speedup.

Computer example

Mathieu Group

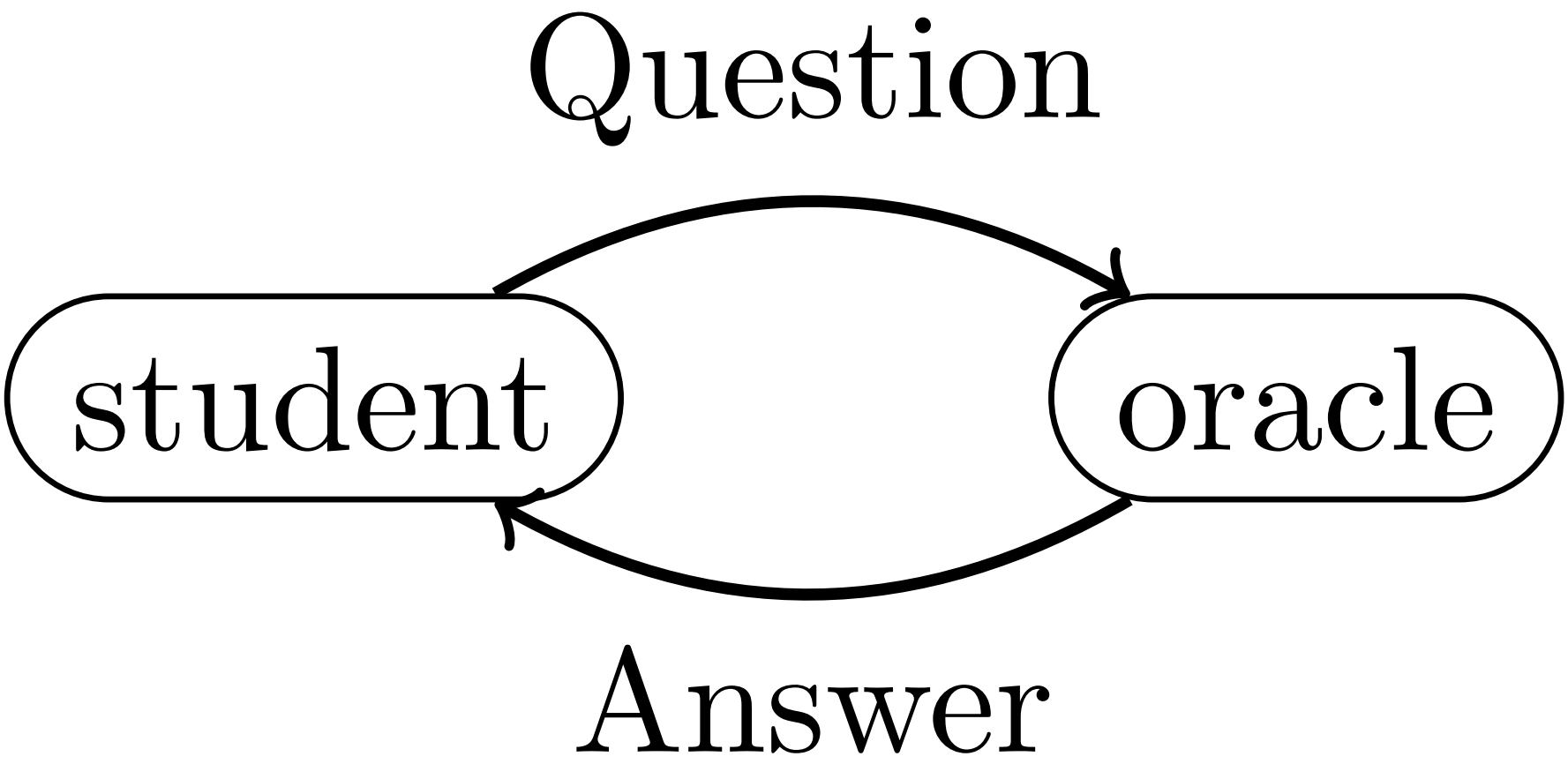
Take $G = M_{12}$ which acts sharply 5-transitively on 12 points. Therefore $b(M_{12}) = 5$. Using GAP, we see that $\gamma = 4$ (ie every irrep of M_{12} appears in the 4th tensor power of the defining rep, and no sooner). So we get a 1-query speedup.

In fact the base size is known to be small (as in ≤ 7) for large classes of permutation groups, including many permutation rep'ns of sporadic simple groups (Burness, Brien, Wilson '07), so we don't get big speedups looking here.

Outline

Learning from an oracle

In a learning task, a student (learner) seeks to determine some hidden information known by an oracle (teacher).



Mathematical examples I

Van Dam's algorithm: identification of a Boolean function with evaluation queries (van Dam '98)

The oracle hides: a Boolean function on n inputs $f : \{1, \ldots, n\} \to \mathbb{Z}_2$.
The student wants to know: What is f?
The student may ask: What is $f(x)$ for some $x \in \{1, \ldots, n\}$?
The oracle answers: $f(x) \in \mathbb{Z}_2$.

Mathematical examples I

Van Dam's algorithm: identification of a Boolean function with evaluation queries (van Dam '98)

The oracle hides: a Boolean function on n inputs $f : \{1, \ldots, n\} \to \mathbb{Z}_2$.
The student wants to know: What is f?
The student may ask: What is $f(x)$ for some $x \in \{1, \ldots, n\}$?
The oracle answers: $f(x) \in \mathbb{Z}_2$.

Special case of GtP!

$\Omega = \{1, \ldots, n\} \times \mathbb{Z}_2$ with $G = \mathbb{Z}_2^n$ acting by

$$f \cdot (x, b) = (x, b \oplus f(x)).$$

As a G-rep, $\mathbb{C}\Omega$ contains the trivial rep and the irreps $-1 \times 1 \times \cdots \times 1, 1 \times -1 \times \cdots \times 1, \ldots$ (ie Hamming weight ≤ 1). Tensor powers give the irreps with higher Hamming weights. Counting the number of sequences by Hamming weight yields a binomial distribution w/parameter n, so we need $n/2 + O(\sqrt{n})$ queries for (bounded error) success (vs n classically).

Mathematical examples II

Bernstein-Vazirani algorithm '97

The oracle hides: an n-bit string $a \in \mathbb{Z}_2^n$.
The student wants to know: What is a?
The student may input: Any n-bit string $x \in \mathbb{Z}_2^n$.
The oracle answers: $x \cdot a = x_1 a_1 + x_2 a_2 + \cdots \in \mathbb{Z}_2$.

Mathematical examples II

Bernstein-Vazirani algorithm '97

The oracle hides: an n-bit string $a \in \mathbb{Z}_2^n$.
The student wants to know: What is a?
The student may input: Any n-bit string $x \in \mathbb{Z}_2^n$.
The oracle answers: $x \cdot a = x_1 a_1 + x_2 a_2 + \cdots \in \mathbb{Z}_2$.

Query complexity

Classically: An optimal algorithm requires n queries to determine f.
Quantumly: The hidden string can be determined (with probability 1) in a single query. (The permutation rep'n for this problem already contains every irrep.)

Mathematical examples III

Many more..

Simon's problem ('95), discrete logarithm (Shor '94), Deutsch-Jozsa problem ('92), PARITY (BBCMdW '98), Grover search ('96), hidden subgroup problem, polynomial interpolation (CvDHS, '16), qudit summation (MP '11, Zhandry '15)...

Mathematical examples III

Many more..

Simon's problem ('95), discrete logarithm (Shor '94), Deutsch-Jozsa problem ('92), PARITY (BBCMdW '98), Grover search ('96), hidden subgroup problem, polynomial interpolation (CvDHS, '16), qudit summation (MP '11, Zhandry '15)...

We're interested in the separation between quantum and classical computing power.

Mathematical examples III

Many more..

Simon's problem ('95), discrete logarithm (Shor '94), Deutsch-Jozsa problem ('92), PARITY (BBCMdW '98), Grover search ('96), hidden subgroup problem, polynomial interpolation (CvDHS, '16), qudit summation (MP '11, Zhandry '15)...

We're interested in the separation between quantum and classical computing power.

Given a learning problem we must

1. Construct algorithms to solve the problem (upper bound on query complexity)

Mathematical examples III

Many more..

Simon's problem ('95), discrete logarithm (Shor '94), Deutsch-Jozsa problem ('92), PARITY (BBCMdW '98), Grover search ('96), hidden subgroup problem, polynomial interpolation (CvDHS, '16), qudit summation (MP '11, Zhandry '15)...

We're interested in the separation between quantum and classical computing power.

Given a learning problem we must

1. Construct algorithms to solve the problem (upper bound on query complexity)
2. Find the minimum number of queries required (lower bound on query complexity).

Quantum learning models.

Quantum setting: oracle hides an unknown unitary operator sampled from a finite set of unitaries.

Quantum learning models.

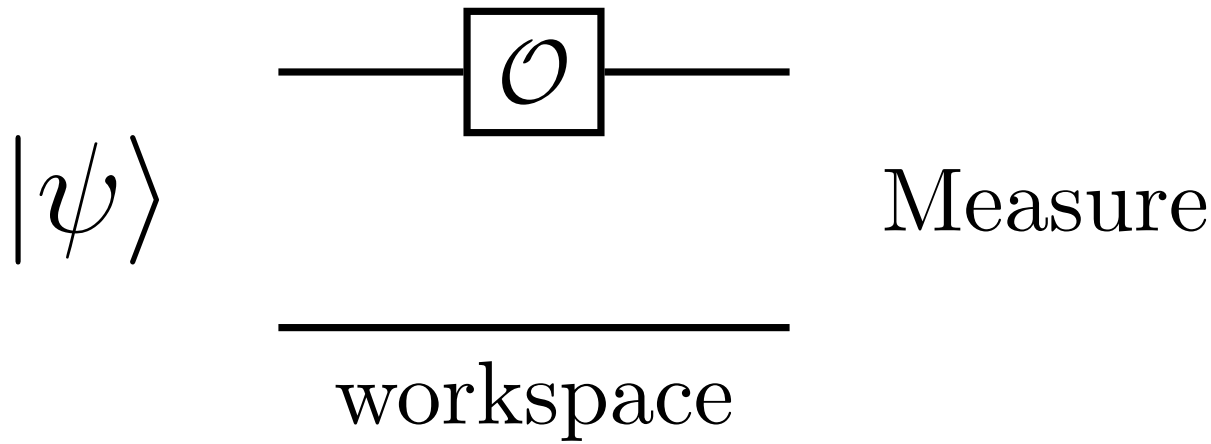
Quantum setting: oracle hides an unknown unitary operator sampled from a finite set of unitaries.

A quantum computer may prepare states (unit vectors in the Hilbert space of the computer), input them to the operator, and perform measurements to deduce information about the hidden unitary.

Quantum learning models.

Quantum setting: oracle hides an unknown unitary operator sampled from a finite set of unitaries.

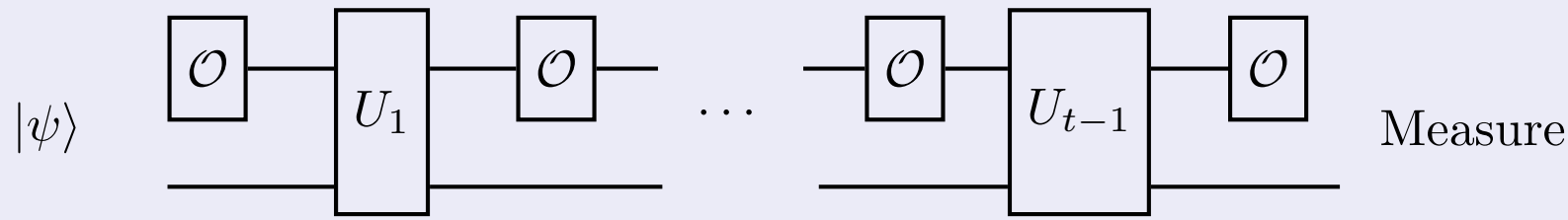
A quantum computer may prepare states (unit vectors in the Hilbert space of the computer), input them to the operator, and perform measurements to deduce information about the hidden unitary.



Single query algorithm to learn $\mathcal{O}$.

Multiple query models.

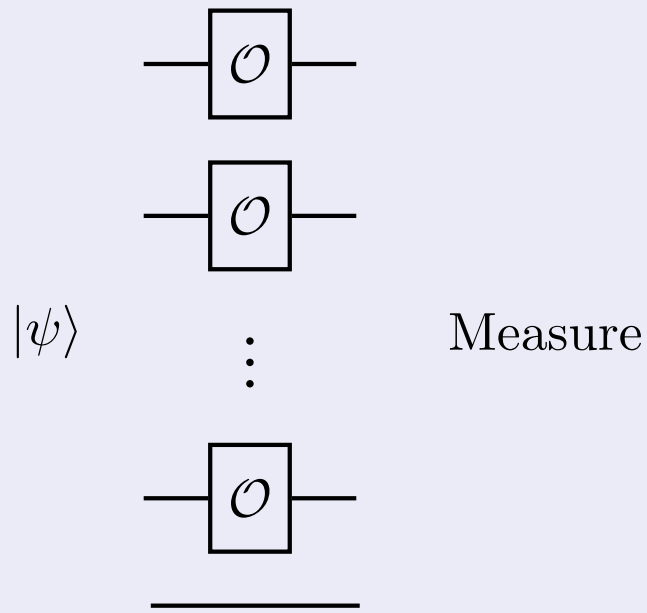
Adaptive queries.



Algorithm designer gets to pick the input $|\psi\rangle$, intermediate unitaries $U_1, \ldots U_{t-1}$, and the measurement.

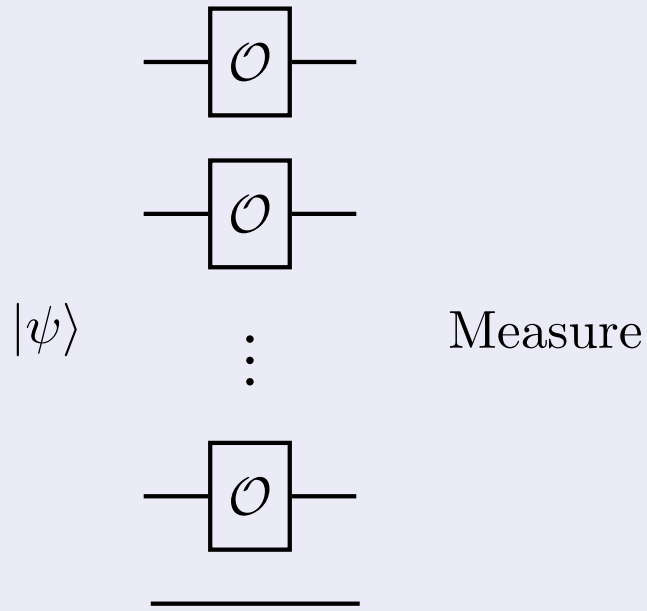
Multiple query models.

Nonadaptive queries.



Multiple query models.

Nonadaptive queries.



A t-query non-adaptive algorithm with access to the oracle $\mathcal{O}$ is exactly the same as a single-query algorithm with access to the oracle $\mathcal{O}^{\otimes t}$!

Symmetric problems use nonadaptive algorithms.

Theorem. (C., Pommersheim)

If the unknown unitaries in a learning problem form a group (ie they form a unitary representation $\pi : G \to U(V)$ of some group G), and the goal of the learning problem is to determine an unknown element $g \in G$, then the optimal success probability of a t-query algorithm is achieved by a non-adaptive algorithm.

Symmetric problems use nonadaptive algorithms.

Theorem. (C., Pommersheim)

If the unknown unitaries in a learning problem form a group (ie they form a unitary representation $\pi : G \to U(V)$ of some group G), and the goal of the learning problem is to determine an unknown element $g \in G$, then the optimal success probability of a t-query algorithm is achieved by a non-adaptive algorithm.

Remark. The actual result concerns a more general learning problem (Coset Identification).

Symmetric problems use nonadaptive algorithms.

Theorem. (C., Pommersheim)

If the unknown unitaries in a learning problem form a group (ie they form a unitary representation $\pi : G \to U(V)$ of some group G), and the goal of the learning problem is to determine an unknown element $g \in G$, then the optimal success probability of a t-query algorithm is achieved by a non-adaptive algorithm.

Remark. The actual result concerns a more general learning problem (Coset Identification).

Theorem.

(Bucicovschi, C., Meyer, Pommersheim 2016) An optimal single-query algorithm to identify an unkown element $g \in G$ succeeds with probability

$$P_{\text{success}} = \frac{1}{|G|} \sum_{\chi} \dim(\chi)^2$$

with the sum taken over all irreps χ which appear in the rep'n V.

Putting these together:

Suppose $\pi : G \to U(V)$ is a unitary rep'n of G, and an oracle hides a unitary operator $\pi(g)$ (with g drawn uniformly at random).

Theorem.

An optimal t-query quantum algorithm successfully identifies the unknown unitary $\pi(g) \in U(V)$ with probability

$$P_{\text{success}} = \frac{1}{|G|} \sum_{\chi} dim(\chi)^2$$

where the sum is over all irreps which appear in $V^{\otimes t}$.

A bunch of new, non-abelian learning problems to study via character theory.

Putting these together:

Suppose $\pi : G \to U(V)$ is a unitary rep'n of G, and an oracle hides a unitary operator $\pi(g)$ (with g drawn uniformly at random).

Theorem.

An optimal t-query quantum algorithm successfully identifies the unknown unitary $\pi(g) \in U(V)$ with probability

$$P_{\text{success}} = \frac{1}{|G|} \sum_{\chi} dim(\chi)^2$$

where the sum is over all irreps which appear in $V^{\otimes t}$.

A bunch of new, non-abelian learning problems to study via character theory.

End

Thank you! Stay tuned for more from Jamie on Saturday!