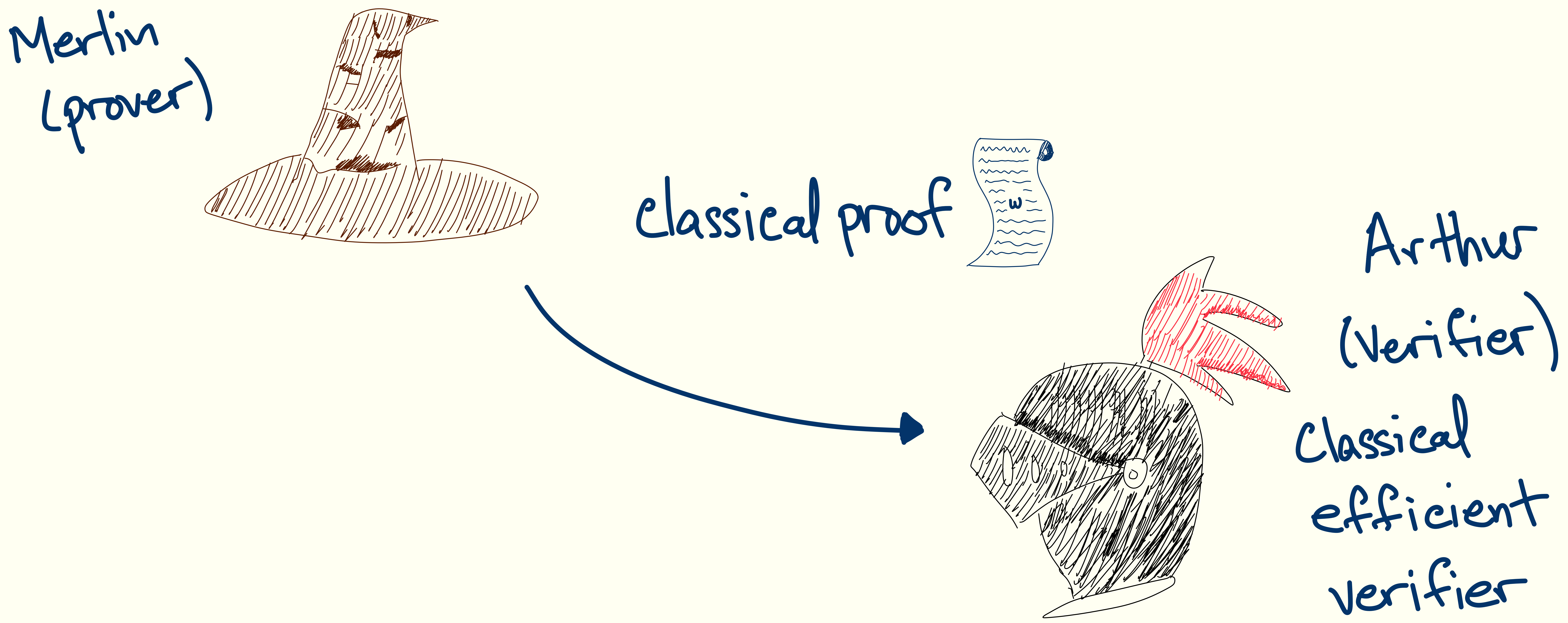
Separating QMA from QCMA with a classical oracle

John Bostanci, Jonas Haferkamp, Chinmay Nirkhe, and Mark Zhandry

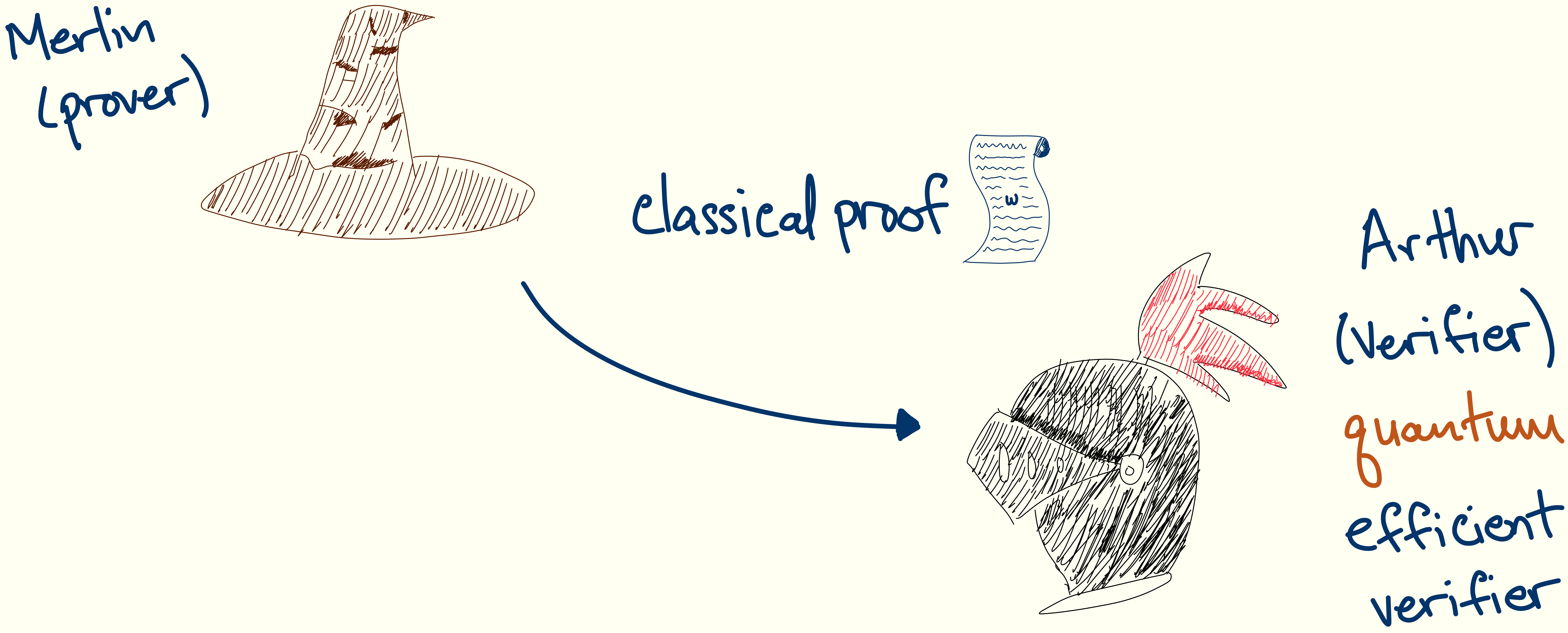
How do model the power of proofs?

In complexity theory, the class NP captures the kinds of problems that we hope to be able to prove to one another.



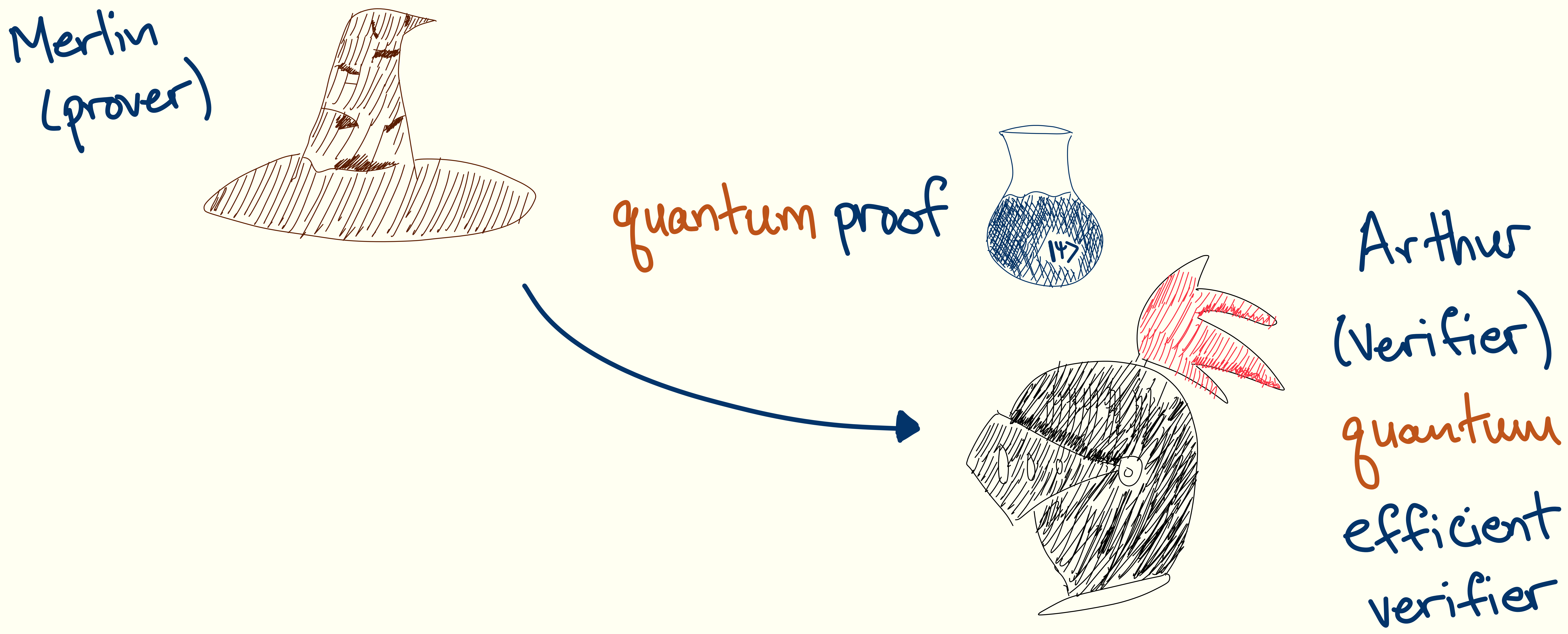
How do model the power of quantum proofs?

With quantum computers, we can compare the relative powers of quantum proofs and classical proofs. QCMA captures the kinds of problems we could prove classically.



How do model the power of quantum proofs?

With quantum computers, we can compare the relative powers of quantum proofs and classical proofs. QMA captures the kinds of problems we could prove quantumly.



Why care about QMA versus QCMA?

The local Hamiltonian problem is QMA-complete.

- Quantum analog of constraint satisfaction.

Why care about QMA versus QCMA?

The local Hamiltonian problem is QMA-complete.

- Quantum analog of constraint satisfaction.

- Captures a lot of "physics"
  - Ground energy of Hamiltonians
  - Representability of Fermionic systems
  - Identity check for quantum channels

Why care about QMA versus QCMA?

The local Hamiltonian problem is QMA-complete.

- Quantum analog of constraint satisfaction.

- Captures a lot of “physics”
  - Ground energy of Hamiltonians
  - Representability of Fermionic systems
  - Identity check for quantum channels

- If QCMA = QMA, then anything you could verify about a ground state could be written down as a classical string!

Why care about QMA versus QCMA?

The local Hamiltonian problem is QMA-complete.

- Quantum analog of constraint satisfaction.

- Captures a lot of "physics"
  - Ground energy of Hamiltonians
  - Representability of Fermionic systems
  - Identity check for quantum channels

- If QCMA = QMA, then anything you could verify about a ground state could be written down as a classical string!
- Otherwise, there must be something interesting about ground states you can only learn from having a copy of the state!

Why care about QMA versus QCMA?

Studying QMA versus QCMA is kind of like asking:

Are all "relevant" properties of quantum ground states possible to write down classically?

What can we say?

We could prove that they are equal… but say that we don’t want to do that.

What can we say?

We could prove that they are equal… but say that we don't want to do that.

Unfortunately, proving an outright separation between the two classes would imply $\mathsf{P} \neq \mathsf{PSPACE}$, among other things.

What can we say?

We could prove that they are equal… but say that we don't want to do that.

Unfortunately, proving an outright separation between the two classes would imply $\mathsf{P} \neq \mathsf{PSPACE}$, among other things.

Next best thing: Oracle separation!

What can we say?

We could prove that they are equal… but say that we don't want to do that.

Unfortunately, proving an outright separation between the two classes would imply $\mathsf{P} \neq \mathsf{PSPACE}$, among other things.

Next best thing: Oracle separation!

- There are many kinds of oracle separations we could prove.
  - Quantum oracle separation: Everyone gets access to a family of unitaries $\{U_n\}_{n=1}^{\infty}$.
  - Classical oracle separation: Everyone gets access to a function $\mathcal{O} : \{0,1\}^* \mapsto \{0,1\}$.

What can we say?

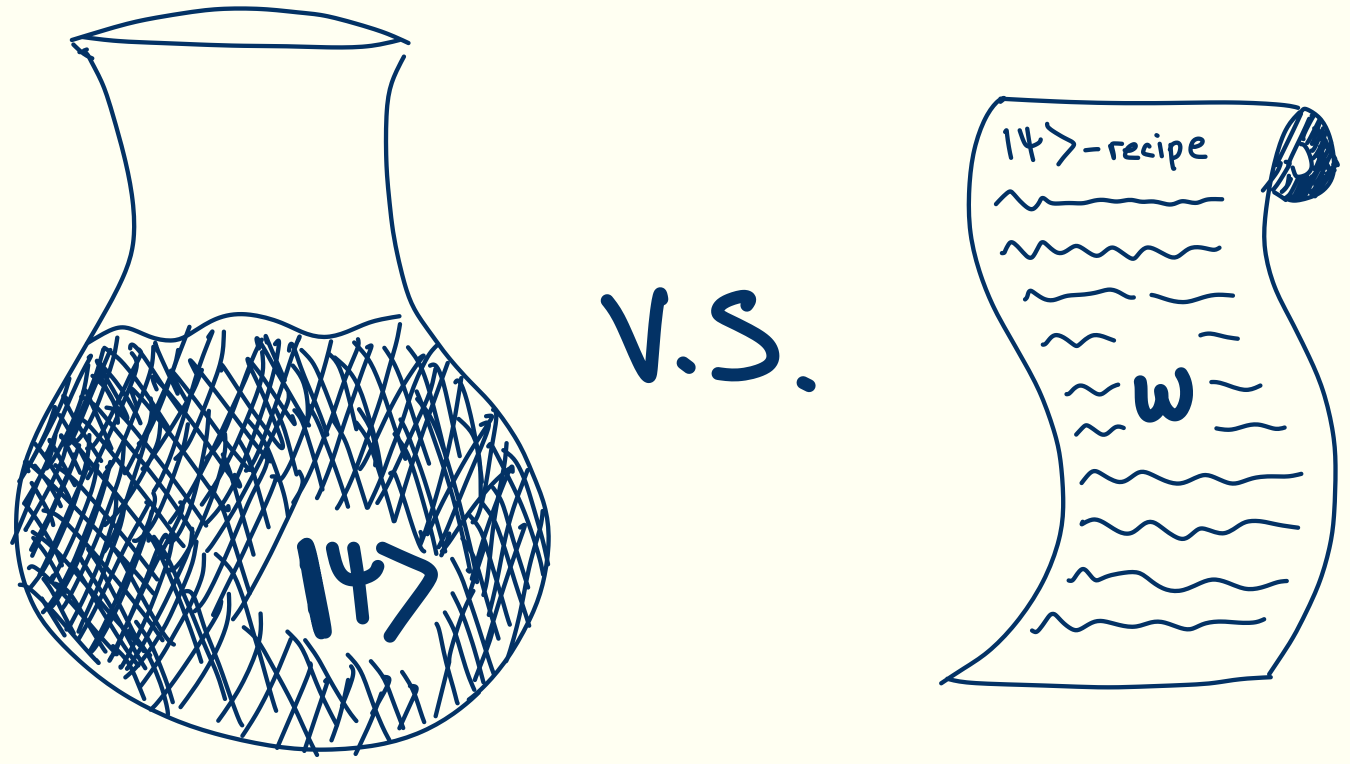
We could prove that they are equal… but say that we don't want to do that.

Unfortunately, proving an outright separation between the two classes would imply $\mathsf{P} \neq \mathsf{PSPACE}$, among other things.

Next best thing: Oracle separation!

- There are many kinds of oracle separations we could prove.
  - Quantum oracle separation: Everyone gets access to a family of unitaries $\{U_n\}_{n=1}^{\infty}$.
  - Classical oracle separation: Everyone gets access to a function $\mathcal{O} : \{0,1\}^* \mapsto \{0,1\}$.
- For this problem, a classical oracle separation is much more challenging (and hopefully interesting) than a quantum oracle separation.

We prove that there is a classical oracle relative to which QMA ≠ QCMA



History of the QMA versus QCMA problem

- First proposed in ’02 by **Aharanov and Naveh.**

History of the QMA **versus** QCMA **problem**

- First proposed in ’02 by **Aharanov and Naveh**.

- **Aaronson & Kuperberg ’06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$

History of the QMA **versus** QCMA **problem**

- First proposed in '02 by **Aharanov and Naveh**.

- **Aaronson & Kuperberg '06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$

- **Lutomirski '11**: Proposed the expander mixing problem as a candidate classical separation.

History of the QMA **versus** QCMA **problem**

- First proposed in ’02 by **Aharanov and Naveh**.

- **Aaronson & Kuperberg ’06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$

- **Lutomirski ’11**: Proposed the expander mixing problem as a candidate classical oracle separation.

- **Fefferman & Kimmel ’15**: In-place permutation oracle. Problem corresponds to set size estimation.

History of the QMA versus QCMA problem

- First proposed in '02 by **Aharanov and Naveh**.
- **Aaronson & Kuperberg '06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$
- **Lutomirski '11**: Proposed the expander mixing problem as a candidate classical oracle separation.
- **Fefferman & Kimmel '15**: In-place permutation oracle. Problem corresponds to set size estimation.
- **Natarajan & Nirkhe '22**: Distribution testing oracle. Problem corresponds to size estimation of an expander graph.

History of the QMA versus QCMA problem

- First proposed in '02 by **Aharanov and Naveh**.
- **Aaronson & Kuperberg '06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$
- **Lutomirski '11**: Proposed the expander mixing problem as a candidate classical oracle separation.
- **Fefferman & Kimmel '15**: In-place permutation oracle. Problem corresponds to set size estimation.
- **Natarajan & Nirkhe '22**: Distribution testing oracle. Problem corresponds to size estimation of an expander graph.
- **Li, Liu, Pelecanos, Yamakawa '23**: Separation assuming only classical queries. Based on "Verifiable Quantum Advantage without Structure".

History of the QMA versus QCMA problem

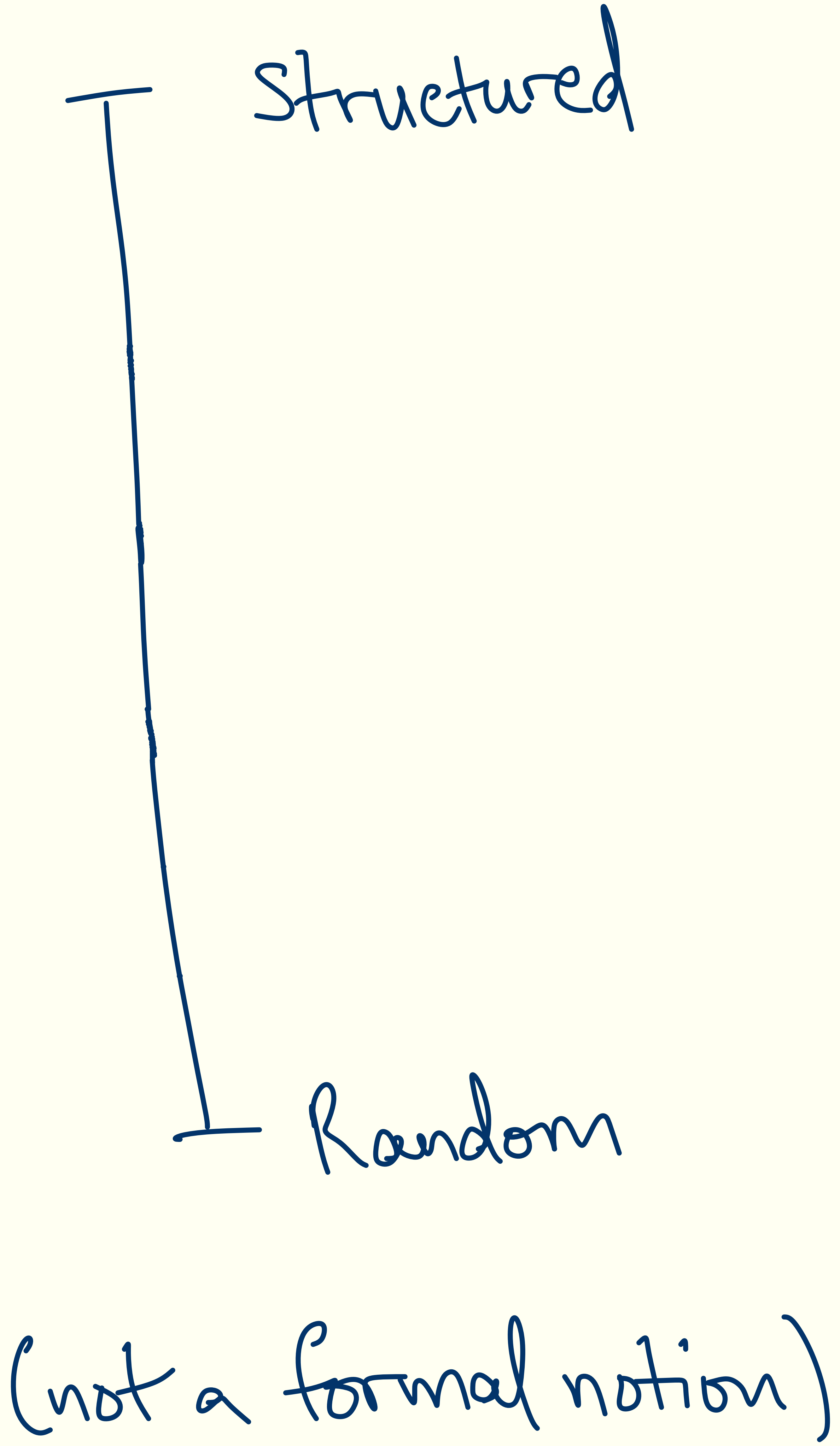
- First proposed in ’02 by **Aharanov and Naveh**.
- **Aaronson & Kuperberg ’06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$
- **Lutomirski ’11**: Proposed the expander mixing problem as a candidate classical oracle separation.
- **Fefferman & Kimmel ’15**: In-place permutation oracle. Problem corresponds to set size estimation.
- **Natarajan & Nirkhe ’22**: Distribution testing oracle. Problem corresponds to size estimation of an expander graph.
- **Li, Liu, Pelecanos, Yamakawa ’23**: Separation assuming only classical queries. Based on “Verifiable Quantum Advantage without Structure”.
- **Ben-David & Kundu ’24**: Bounded adaptivity, based on “Verifiable Quantum Advantage without Structure”.

History of the QMA versus QCMA problem

- First proposed in '02 by **Aharanov and Naveh**.
- **Aaronson & Kuperberg '06**: Quantum oracle separation. Each $\mathcal{O}_n = \mathrm{id} - 2|\psi_n\rangle\langle\psi_n|$
- **Lutomirski '11**: Proposed the expander mixing problem as a candidate classical oracle separation.
- **Fefferman & Kimmel '15**: In-place permutation oracle. Problem corresponds to set size estimation.
- **Natarajan & Nirkhe '22**: Distribution testing oracle. Problem corresponds to size estimation of an expander graph.
- **Li, Liu, Pelecanos, Yamakawa '23**: Separation assuming only classical queries. Based on "Verifiable Quantum Advantage without Structure".
- **Ben-David & Kundu '24**: Bounded adaptivity, based on "Verifiable Quantum Advantage without Structure".
- **Liu, Mutreja, Yuen '25**: Aaronson-Ambainis-like conjecture implies QMA QCMA separation. Problem corresponds to size estimation of an expander graph.

Why is this problem so hard?

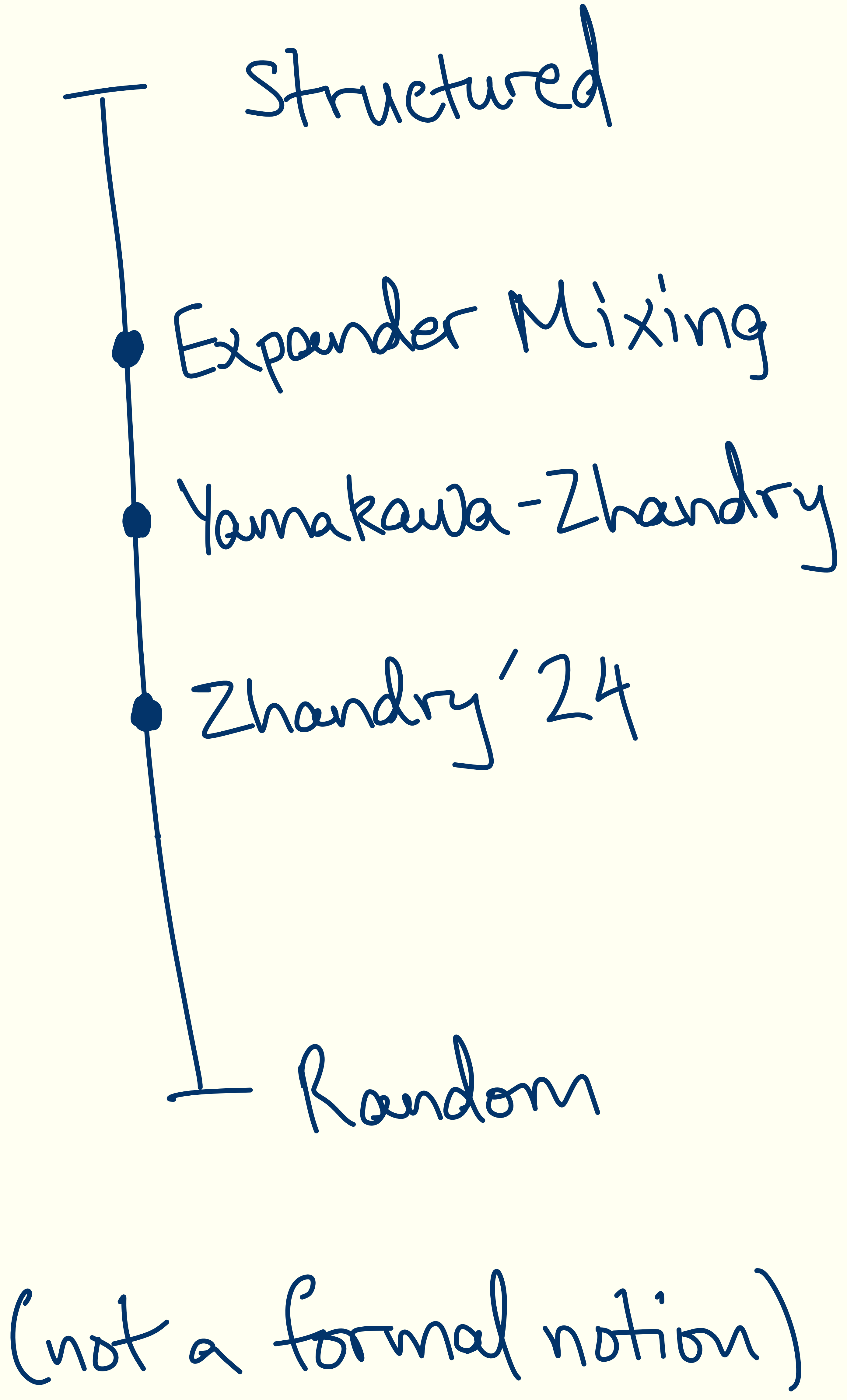
To me, the problem has been "stuck" in between two competing desires for a while.



Why is this problem so hard?

To me, the problem has been "stuck" in between two competing desires for a while.

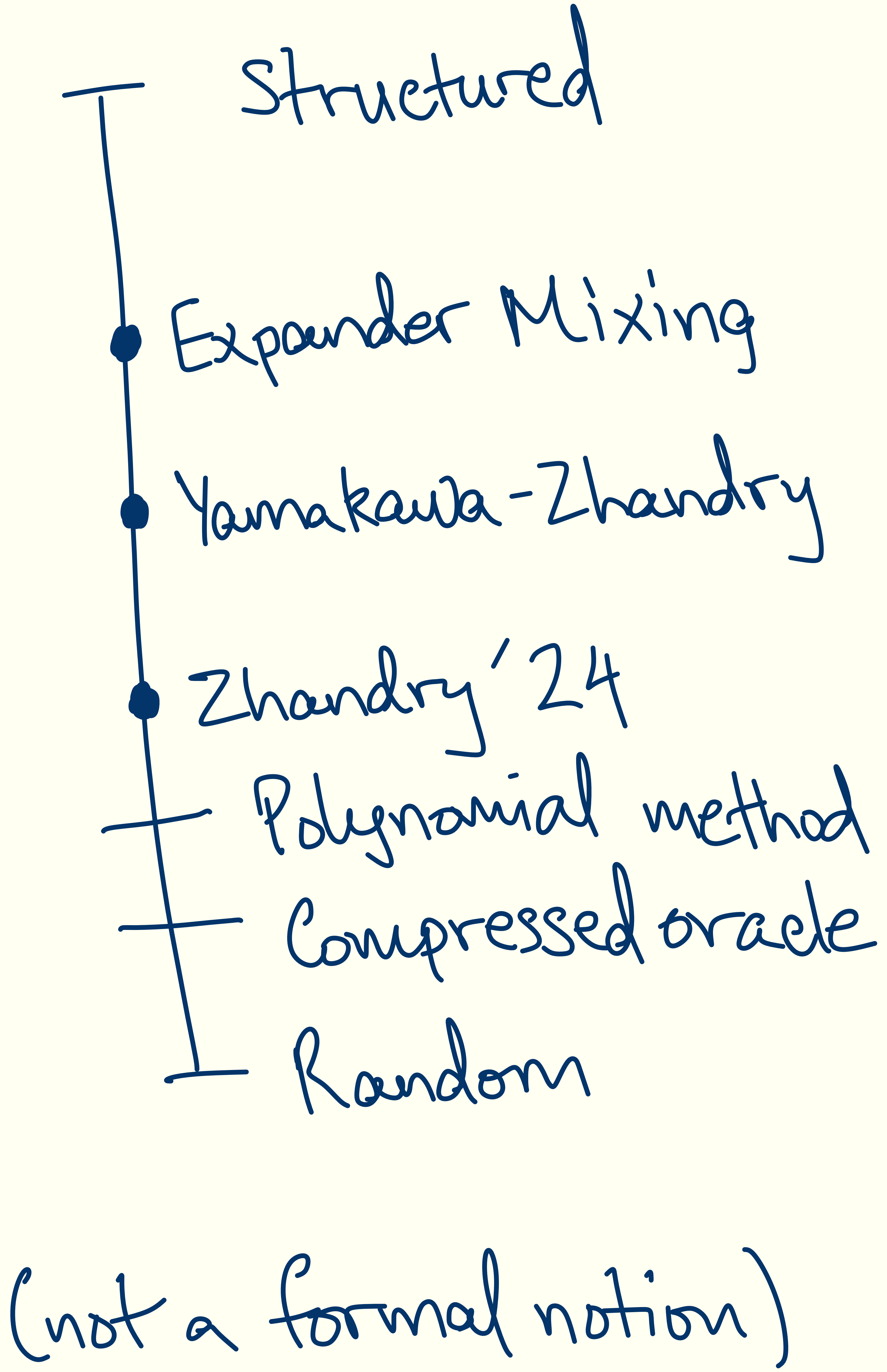
- In any separation, the QMA must do more than just measure their quantum state
  → The classical oracle should have some hidden structure that is only "visible" to a quantum proof.



Why is this problem so hard?

To me, the problem has been "stuck" in between two competing desires for a while.

- In any separation, the QMA must do more than just measure their quantum state
  → The classical oracle should have some hidden structure that is only "visible" to a quantum proof.
- But, quantum lower bound techniques usually take advantage of the randomness of the oracle.
  → Need a new technique for analyzing some kind of structured classical oracles.

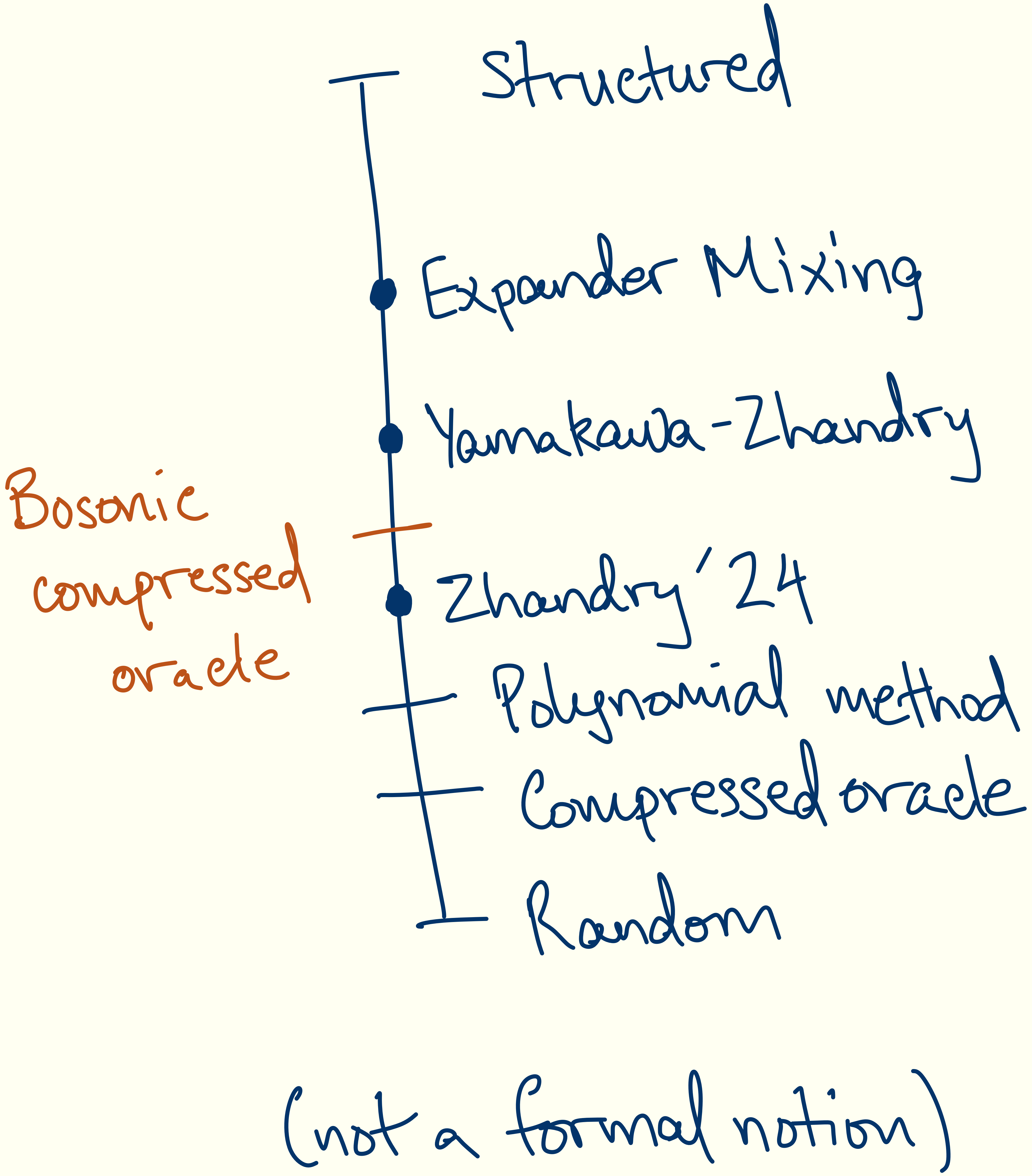


Why is this problem so hard?

To me, the problem has been "stuck" in between two competing desires for a while.

- In any separation, the QMA must do more than just measure their quantum state
  → The classical oracle should have some hidden structure that is only "visible" to a quantum proof.
- But, quantum lower bound techniques usually take advantage of the randomness of the oracle.
  → Need a new technique for analyzing some kind of structured classical oracles.

Our paper bridges the gap, taking the less structured oracle of Zhandry'24, and introducing new analysis.



Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.
- Our input is now a $N = 2^n$-bit string that we treat as an oracle.

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.
- Our input is now a $N = 2^n$-bit string that we treat as an oracle.
- Our verifier get a poly(n)-bit quantum state, and gets to make controlled quantum queries to the oracle.

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.
- Our input is now a $N = 2^n$-bit string that we treat as an oracle.
- Our verifier get a poly(n)-bit quantum state, and gets to make controlled quantum queries to the oracle.
- Has to decide whether the oracle is in L_{yes} or L_{no} with $2/3,\ 1/3$ gap, promised one is the case.

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.
- Our input is now a $N = 2^n$-bit string that we treat as an oracle.
- Our verifier get a poly(n)-bit quantum state, and gets to make controlled quantum queries to the oracle.
- Has to decide whether the oracle is in L_{yes} or L_{no} with $2/3,\ 1/3$ gap, promised one is the case.

Similarly we can define a scaled-down version of a QCMA verifier.

Preliminaries: Oracle input problems

Consider a scaled-down version of a QMA verifier that runs in polylog(N) time:

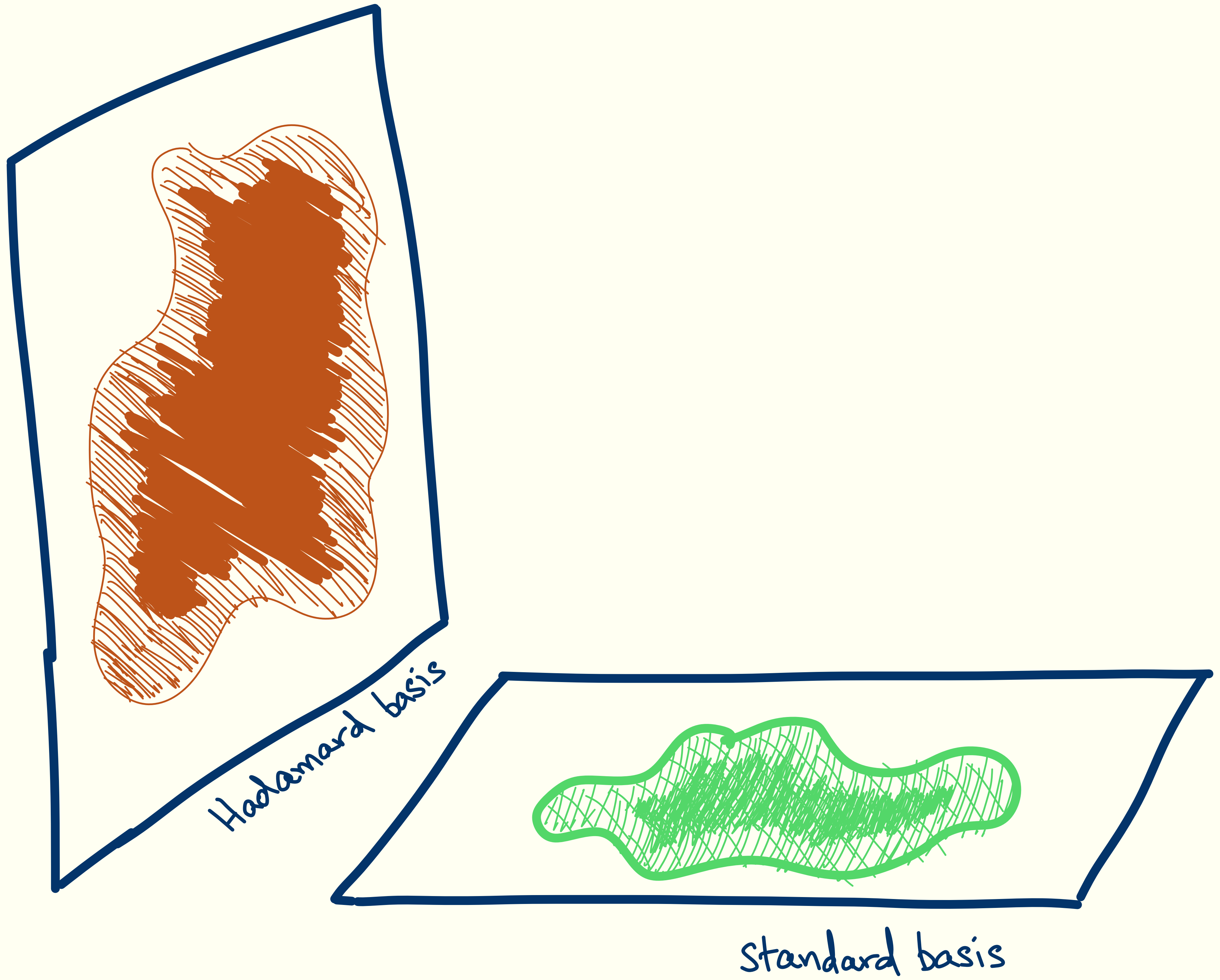
- There is some language $L = (L_{\text{yes}}, L_{\text{no}})$ of N-bit strings.
- Our input is now a $N = 2^n$-bit string that we treat as an oracle.
- Our verifier get a poly(n)-bit quantum state, and gets to make controlled quantum queries to the oracle.
- Has to decide whether the oracle is in L_{yes} or L_{no} with $2/3,\ 1/3$ gap, promised one is the case.

Similarly we can define a scaled-down version of a QCMA verifier.

If you can prove that there is a language separating scaled-down QMA from QCMA, you can use standard diagonalization tricks to turn this into a classical oracle separation.

The spectral Forrelation problem

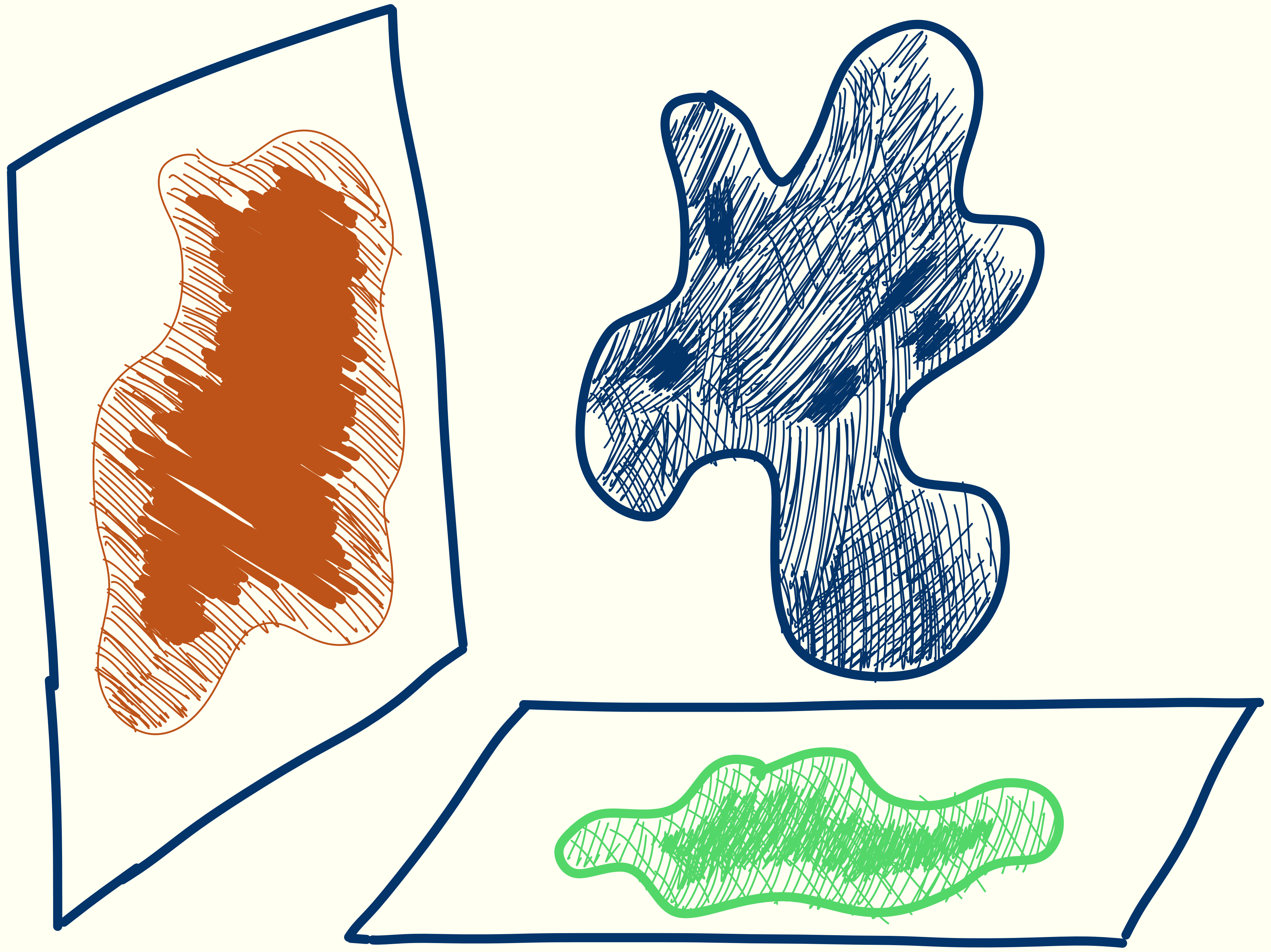
The spectral Forrelation problem is a problem about pairs of sets (S, U), which we treat as oracles through the set membership functions. S~positions, and U~momentums.



The spectral Forrelation problem

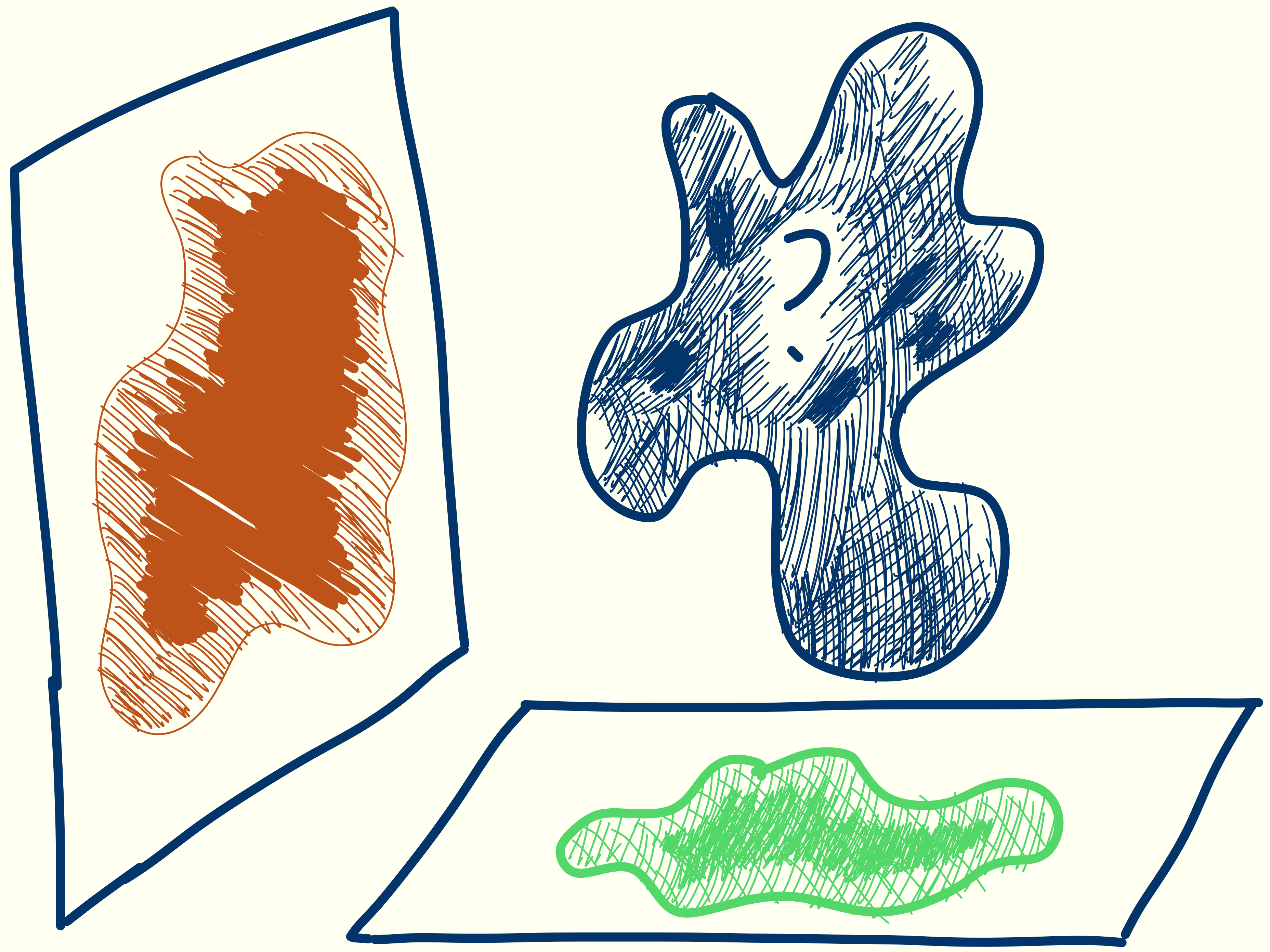
We say that two sets (S, U) are α-spectrally Forrelated if there is a state $|\psi\rangle$ such that

$$\|\Pi_U \cdot H^{\otimes n} \cdot \Pi_S |\psi\rangle\|^2 \geq \alpha$$



The spectral Forrelation problem

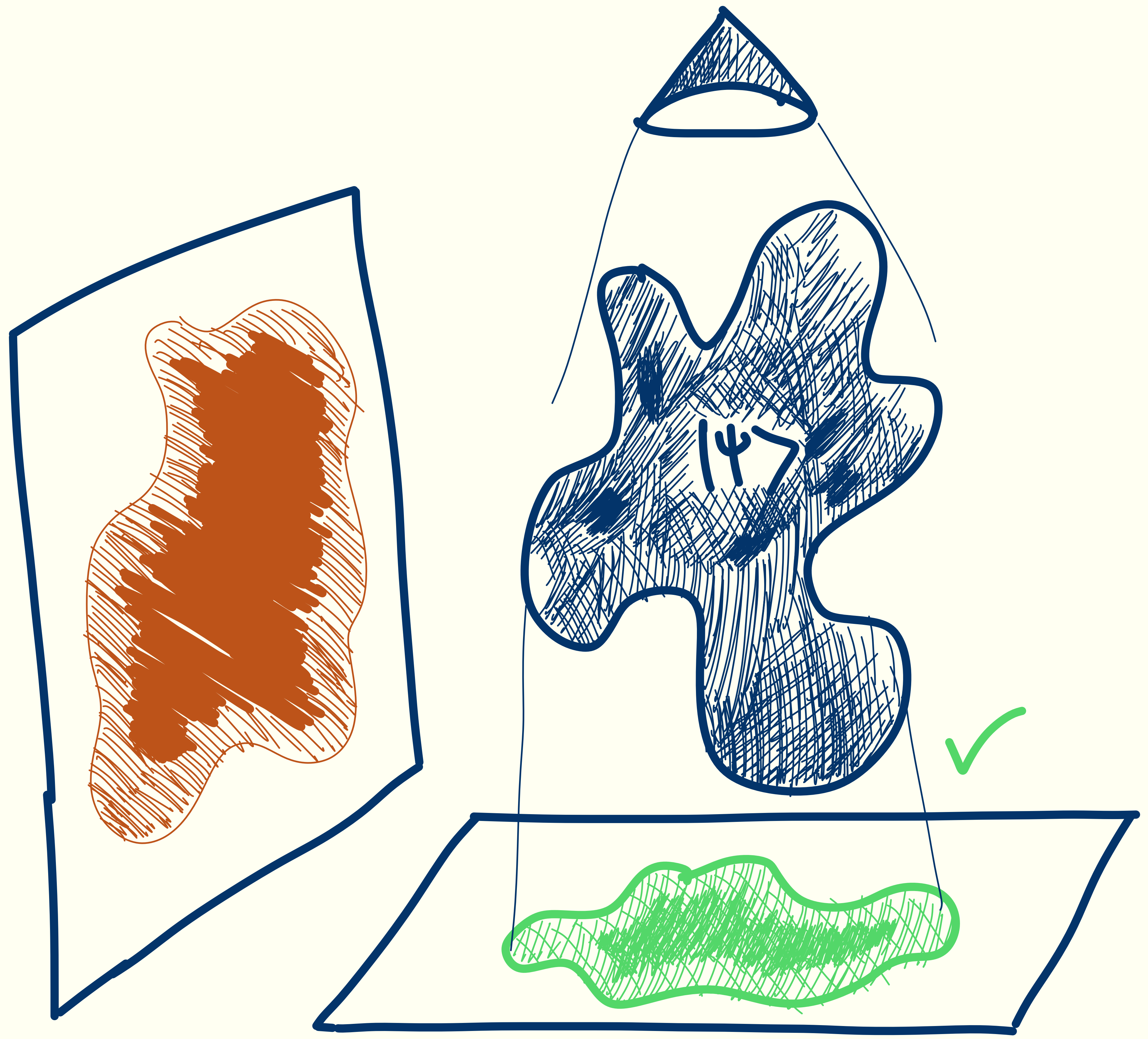
Given oracle access to two sets (S, U) (via set membership functions), determine if there is a state $|\psi\rangle$ such that $\|\Pi_U \cdot H^{\otimes n} \cdot \Pi_S |\psi\rangle\|^2$ is large ($\geq 59/100$) or small ($\leq 57/100$), promised that one of the two is the case.



Spectral Forrelation is in QMA

Given a copy of a state $|\psi\rangle$:

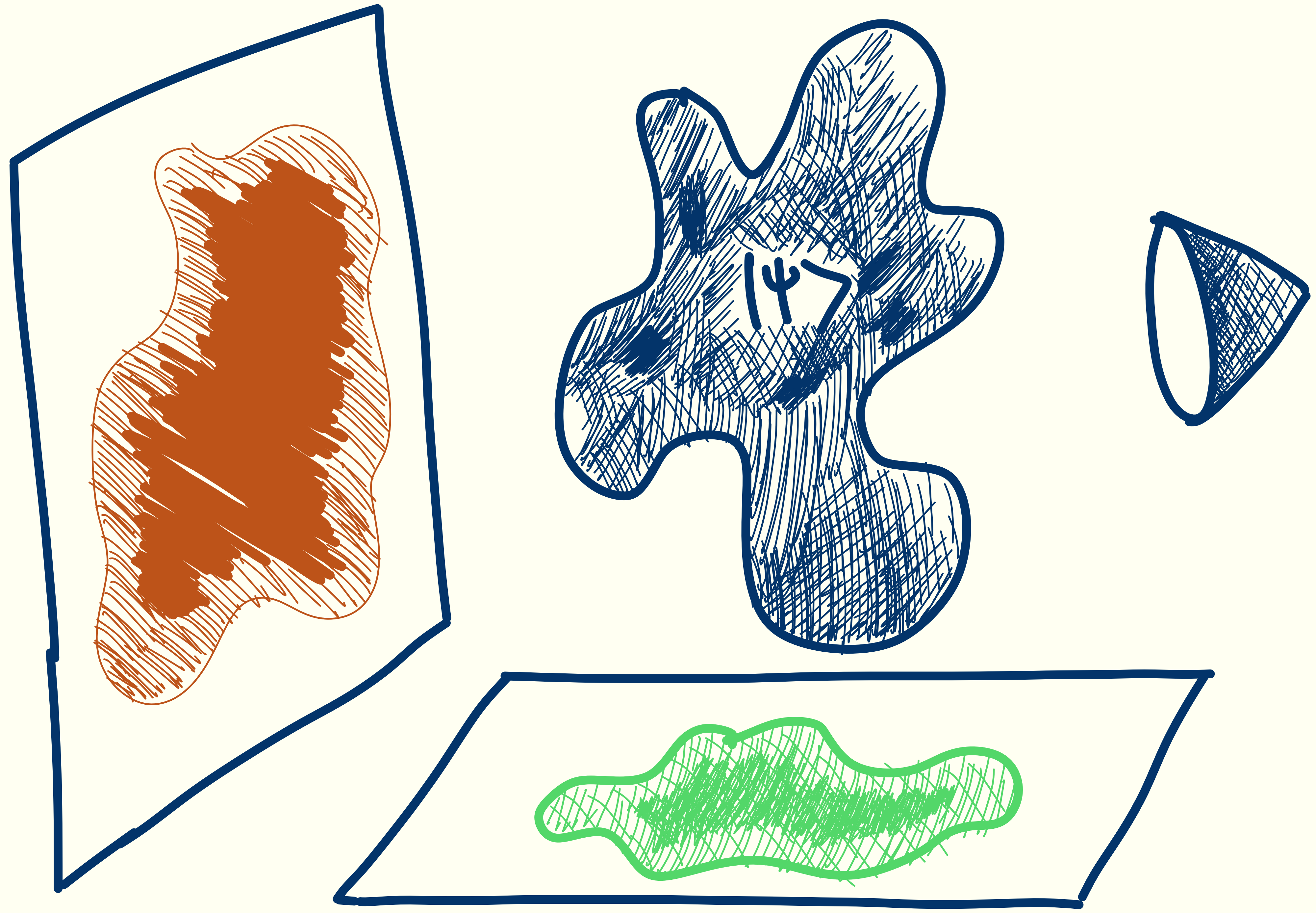
- Use S oracle to measure the POVM $\{\Pi_S, \mathrm{id} - \Pi_S\}$, reject if the outcome is $\mathrm{id} - \Pi_S$.



Spectral Forrelation is in QMA

Given a copy of a state $|\psi\rangle$:

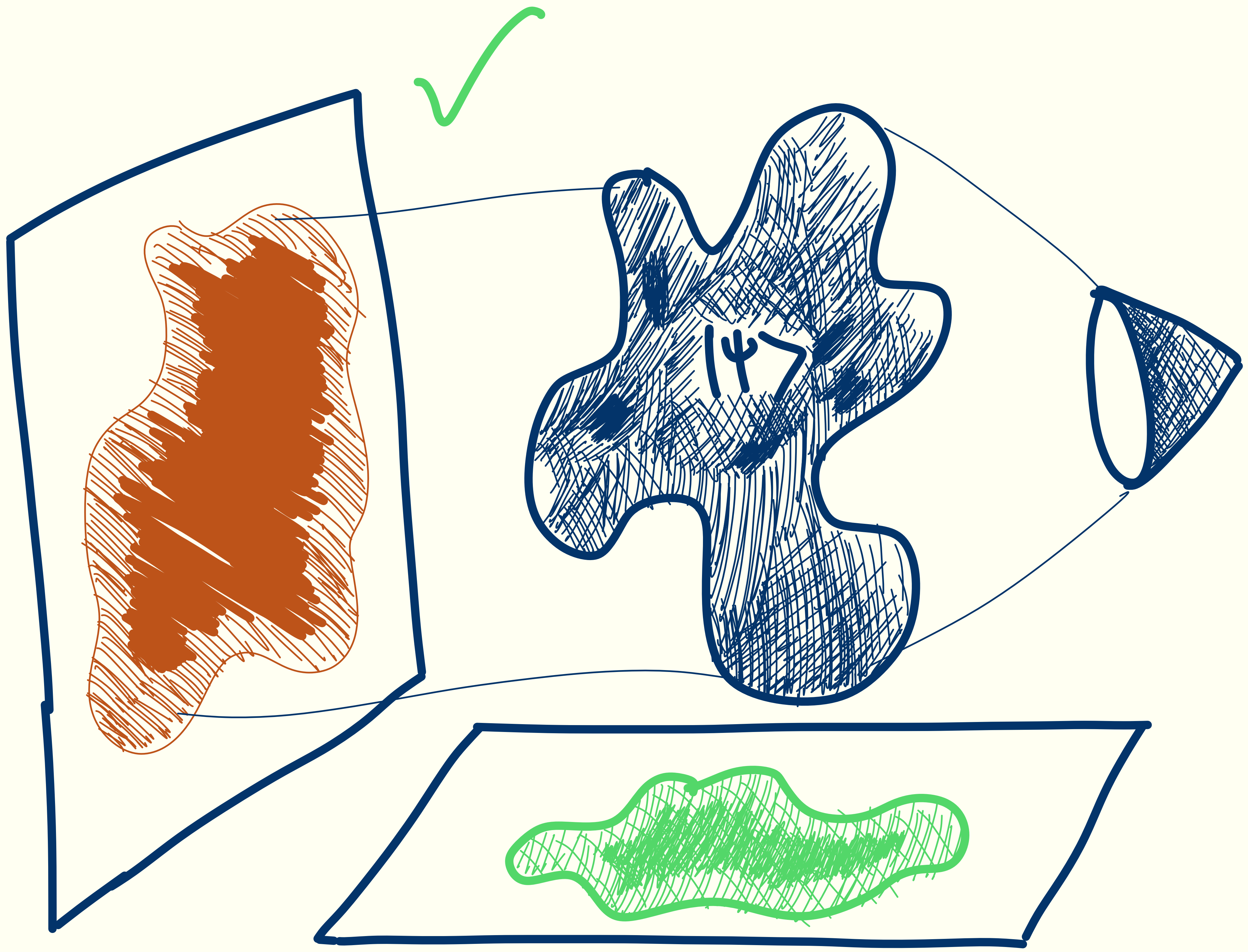
- Use S oracle to measure the POVM $\{\Pi_S, \mathrm{id} - \Pi_S\}$, reject if the outcome is $\mathrm{id} - \Pi_S$.
- Apply $H^{\otimes n}$ to the resulting state.



Spectral Forrelation is in QMA

Given a copy of a state $|\psi\rangle$:

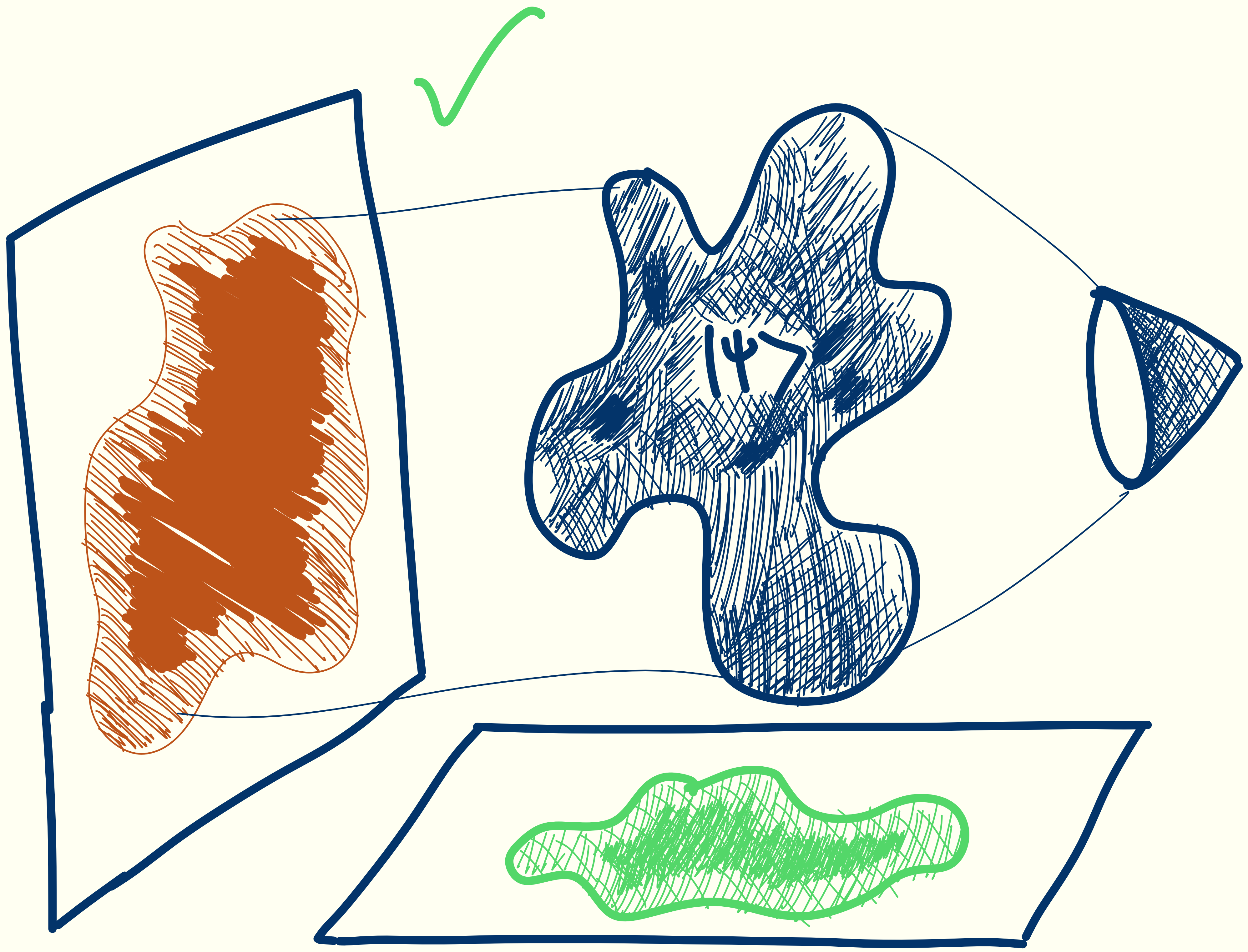
- Use S oracle to measure the POVM $\{\Pi_S, \mathrm{id} - \Pi_S\}$, reject if the outcome is $\mathrm{id} - \Pi_S$.
- Apply $H^{\otimes n}$ to the resulting state.
- Use U oracle to measure the POVM $\{\Pi_U, \mathrm{id} - \Pi_U\}$, reject it the outcome is $\mathrm{id} - \Pi_U$.
- Accept.



Spectral Forrelation is in QMA

Given a copy of a state $|\psi\rangle$:

- Use S oracle to measure the POVM $\{\Pi_S, \text{id} - \Pi_S\}$, reject if the outcome is $\text{id} - \Pi_S$.
- Apply $H^{\otimes n}$ to the resulting state.
- Use U oracle to measure the POVM $\{\Pi_U, \text{id} - \Pi_U\}$, reject it the outcome is $\text{id} - \Pi_U$.
- Accept.

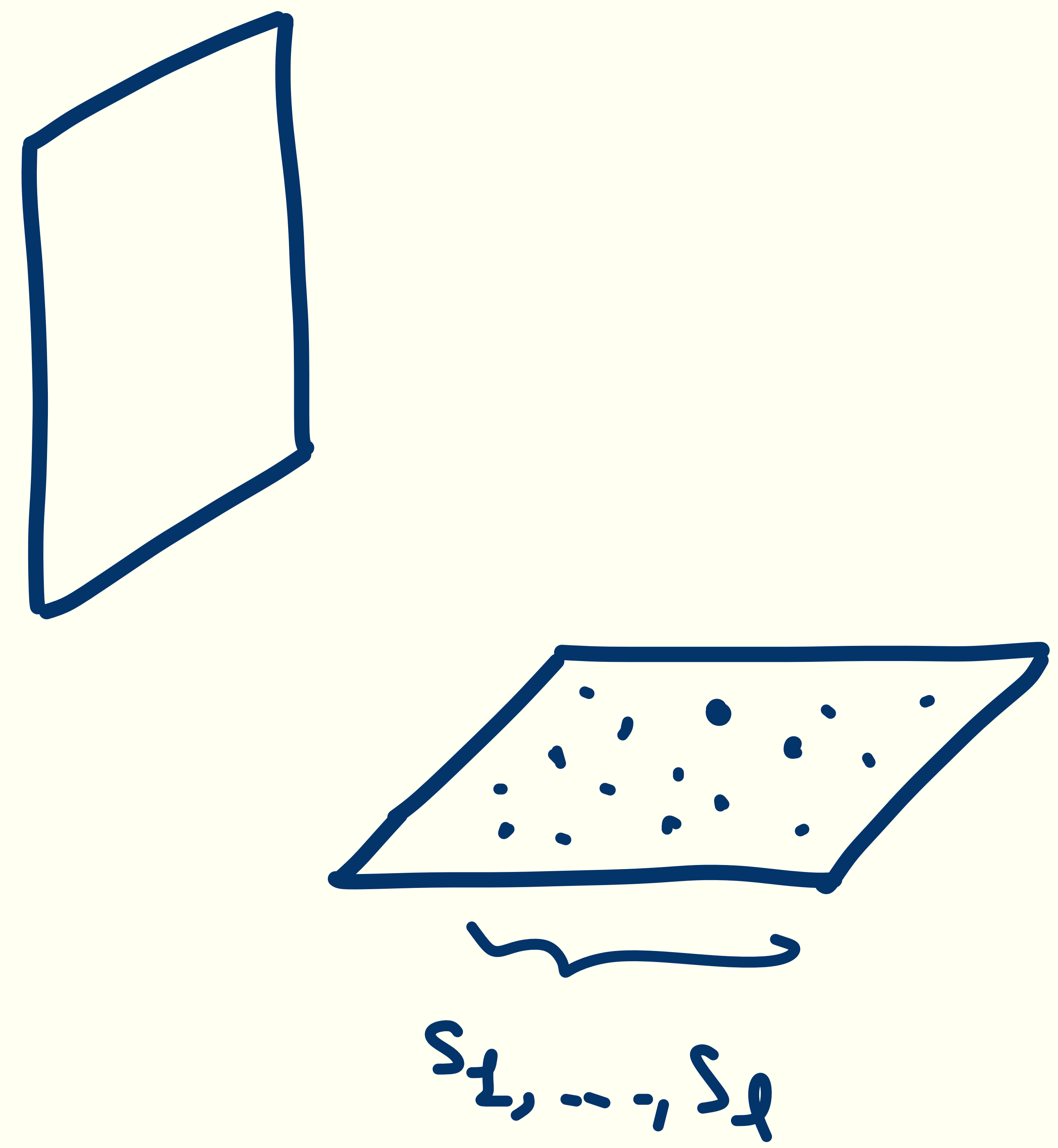


This verifier accepts with probability: $\|\Pi_U \cdot H^{\otimes n} \cdot \Pi_S |\psi\rangle\|^2$.

Marriot-Watrous amplification can bring this to the standard 2/3 or 1/3.

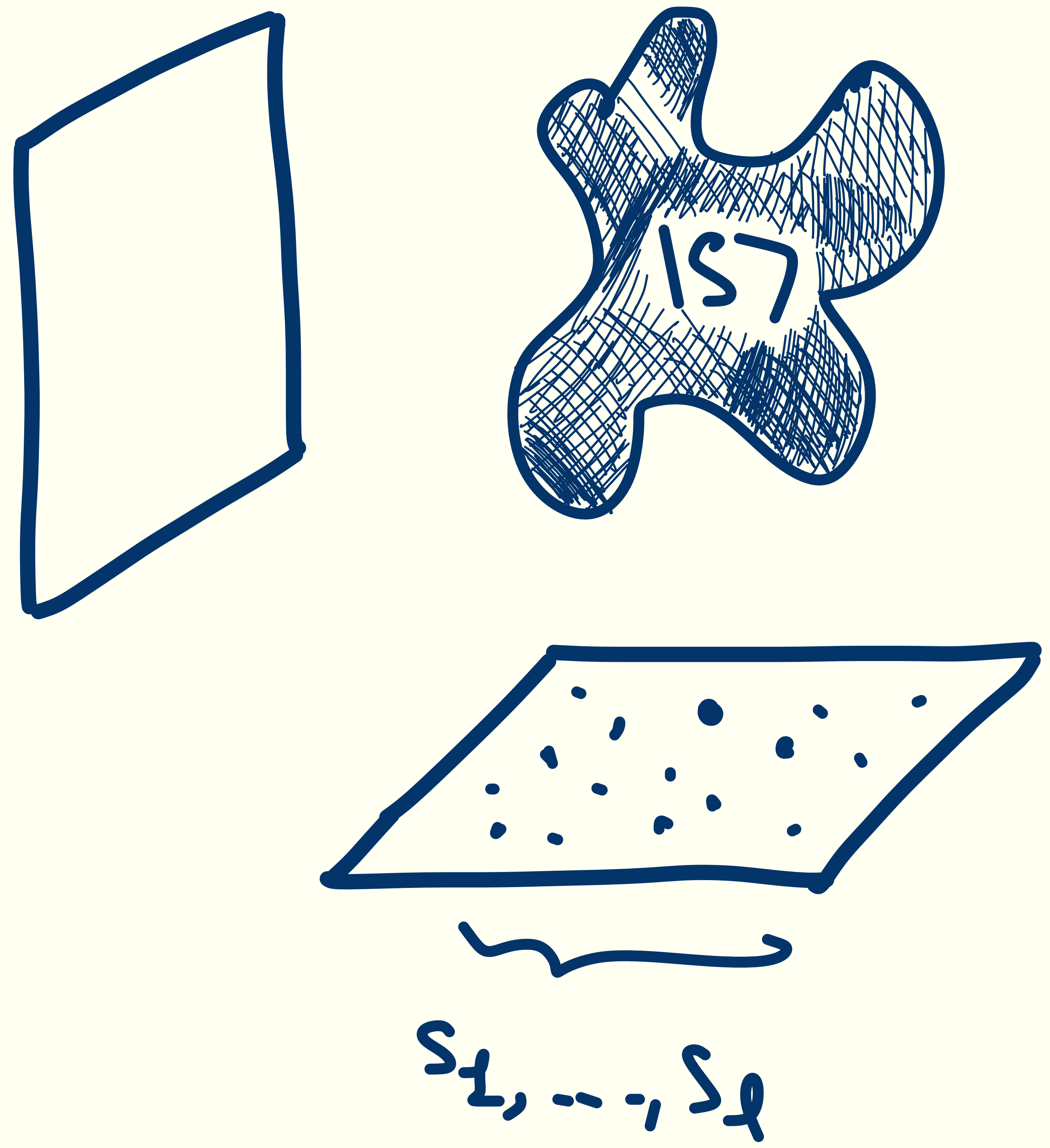
A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$.



A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

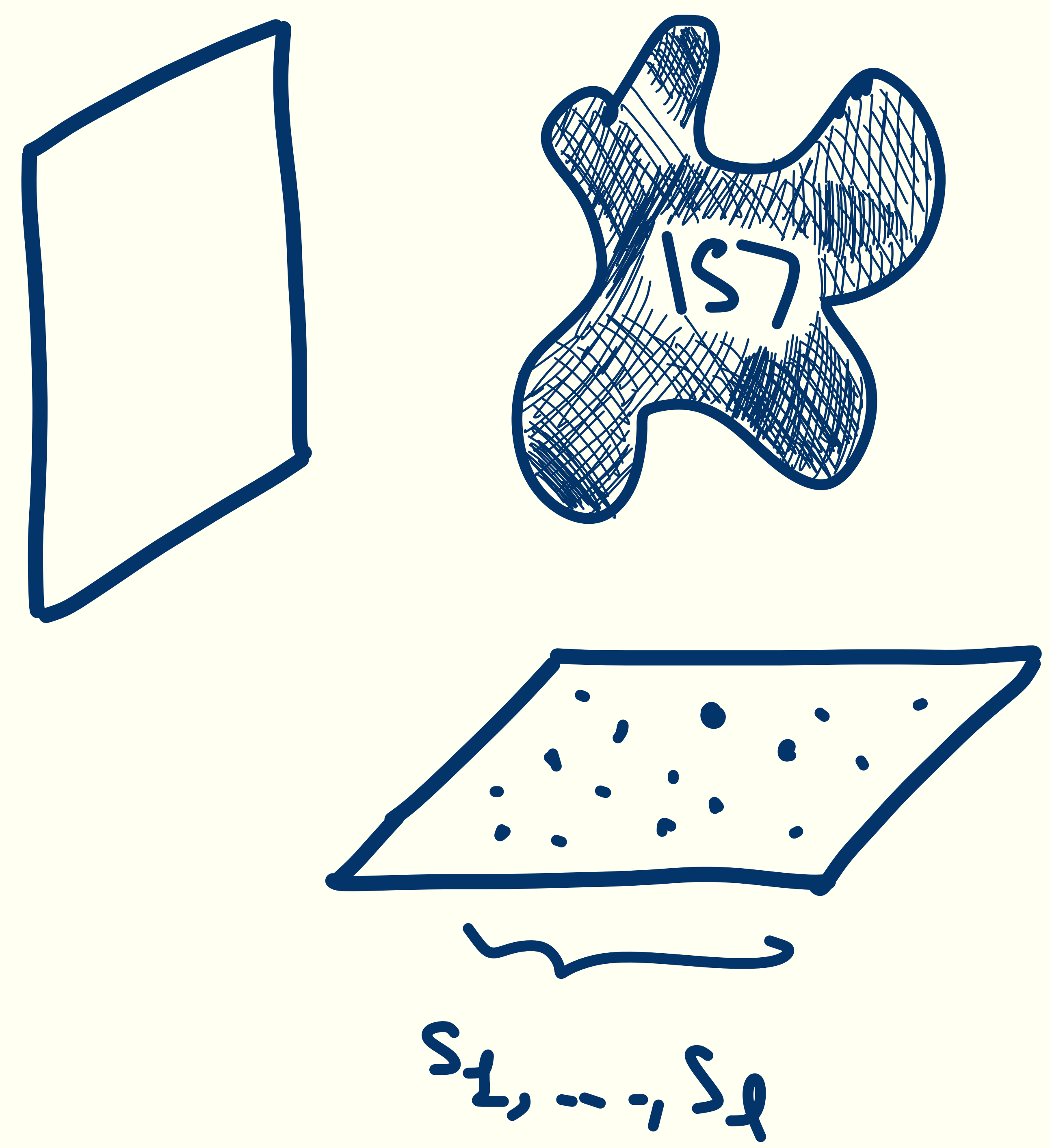


A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}2^n \left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$

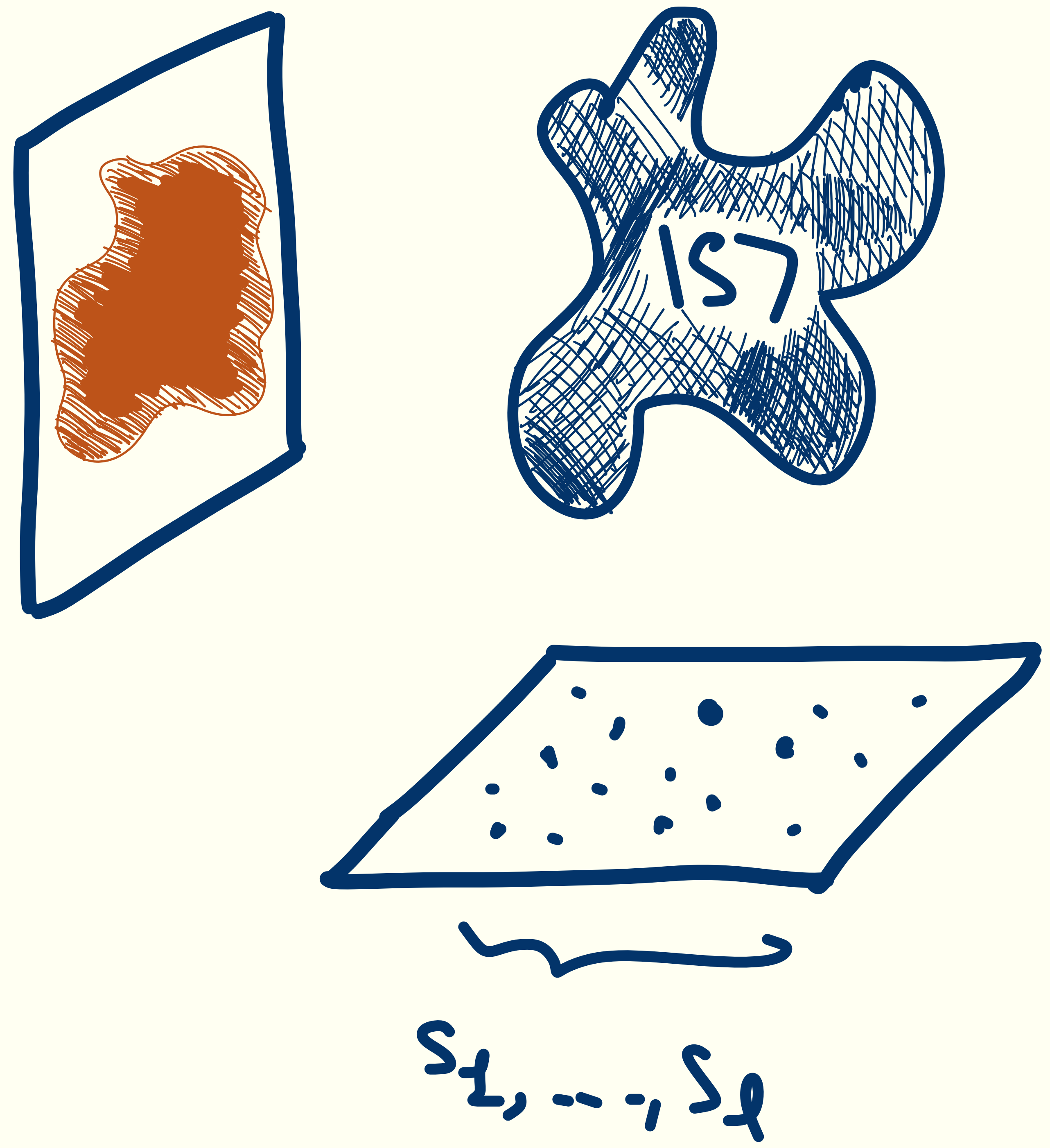


A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2} \exp\left(-\frac{1}{10} 2^n \left| \langle y | H^{\otimes n} | S \rangle \right|^2 \right)$$

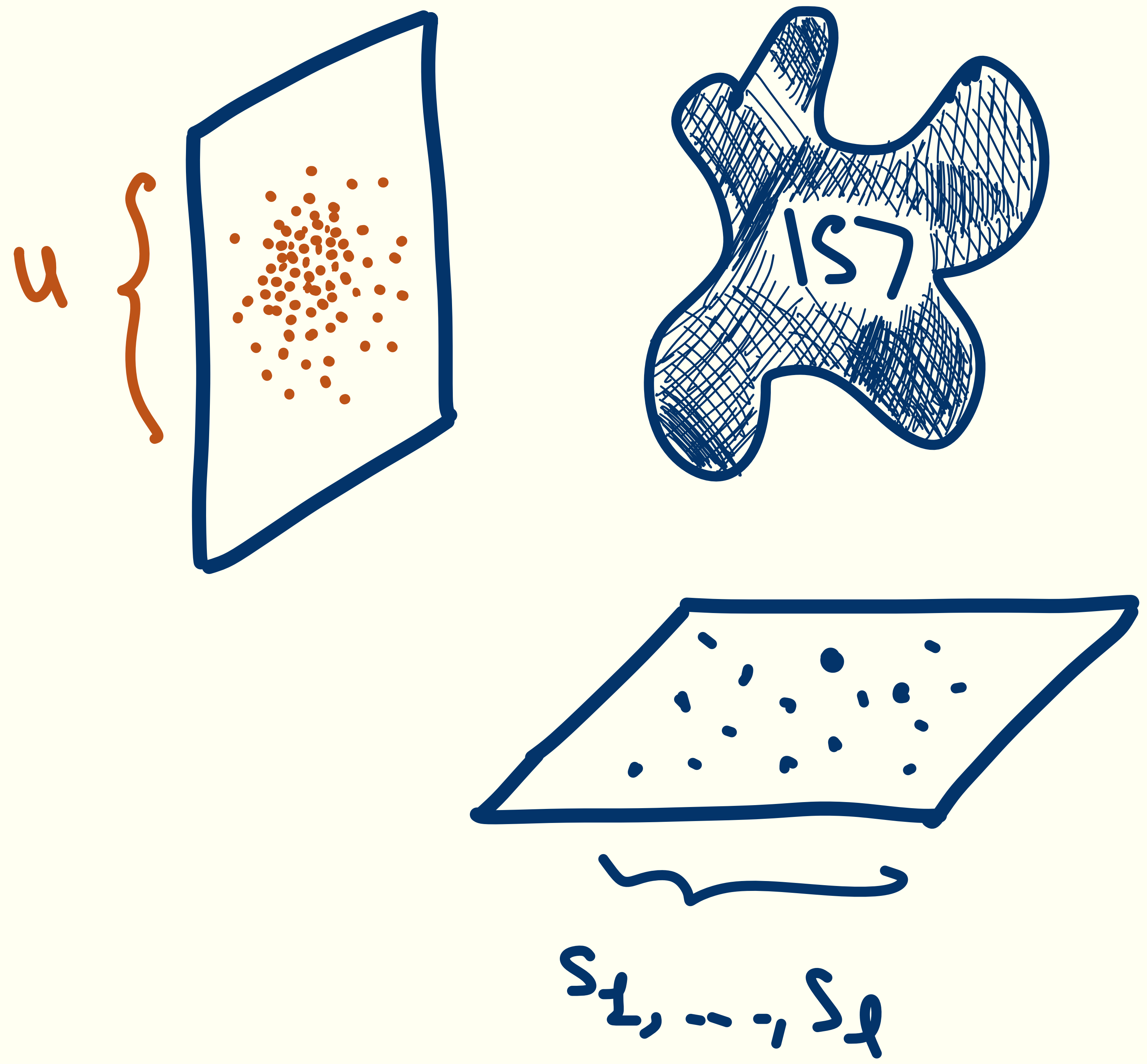


A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}2^n \left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$

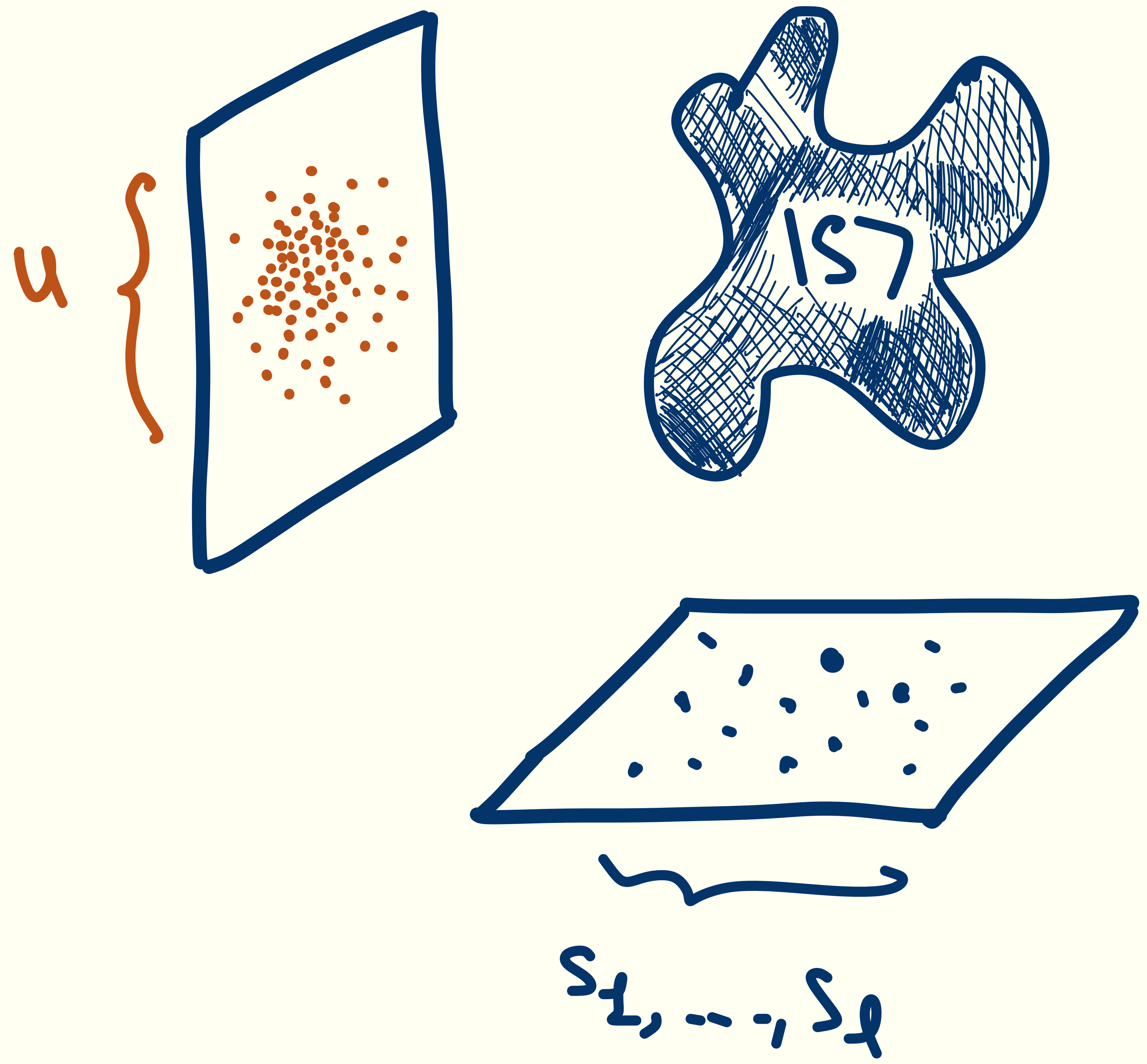


A distribution over spectral Forrelation yes instances

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}2^n \left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$



We call this distribution over oracles the Strong distribution.

A distribution over spectral Forrelation yes instances

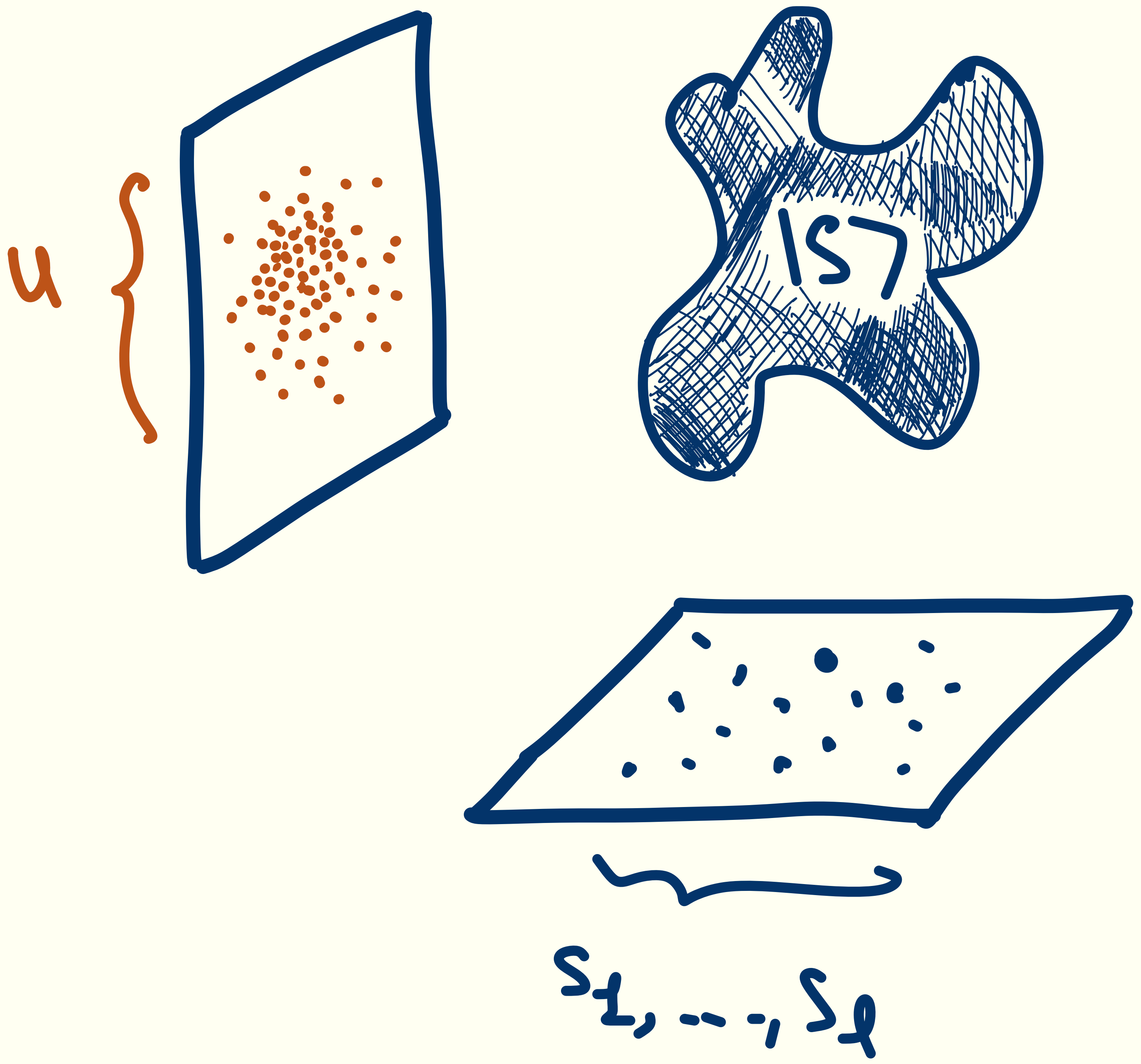
- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}2^n \left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$

↑ $\gamma_y^{(S)}$

We call this distribution over oracles the Strong distribution.



Main theorems

Main theorems

Theorem 1: For all $v > 0$, and all quantum query algorithms making $T = T(n)$ queries to a set membership oracle for U, the probability, over Strong, that the algorithm outputs v distinct points from S is at most

$$\leq \left(\frac{\mathrm{poly}(v, T)}{\mathrm{poly}(2^n)}\right)^v.$$

Main theorems

Theorem 1: For all $v > 0$, and all quantum query algorithms making $T = T(n)$ queries to a set membership oracle for U, the probability, over Strong, that the algorithm outputs v distinct points from S is at most

$$\leq \left(\frac{\mathrm{poly}(v, T)}{\mathrm{poly}(2^n)}\right)^v.$$

Theorem 2: If there exists a QCMA algorithm, making $t = t(n)$ queries to (S, U) and taking a witness of length $q = q(n)$, then for all $0 < v < \ell/100$, there is a query algorithm making vt queries to U that outputs v distinct points from S with probability

$$\geq 2^{-q}\left(\frac{1}{36t^2}\right)^v$$

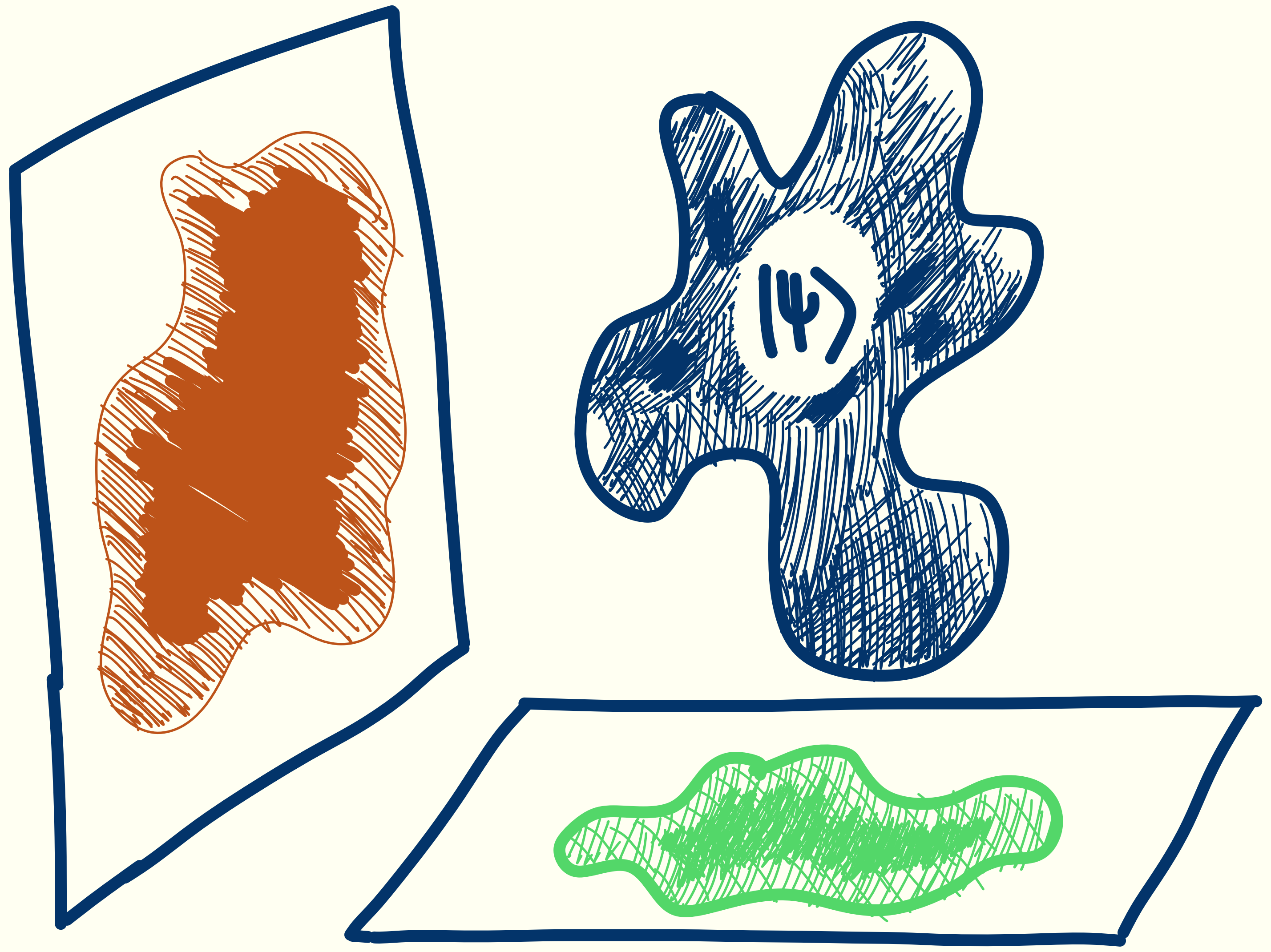
Strong yes instances

A pair (S, U) is a strong yes instance if:

Strong yes instances

A pair (S, U) is a strong yes instance if:

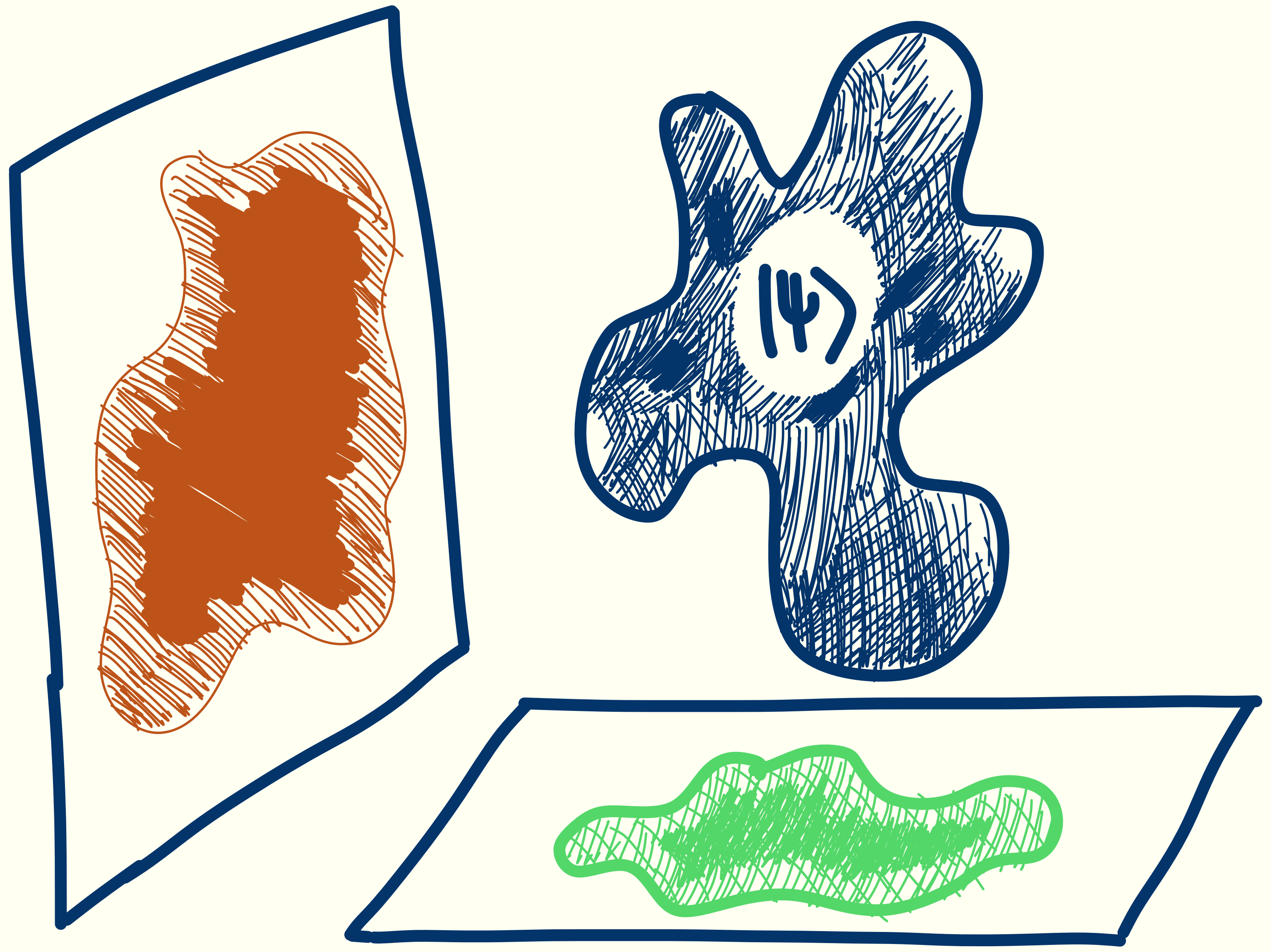
- (S, U) is a yes instance of spectral Forrelation (i.e., $\geq 59/100$ spectrally Forrelated).

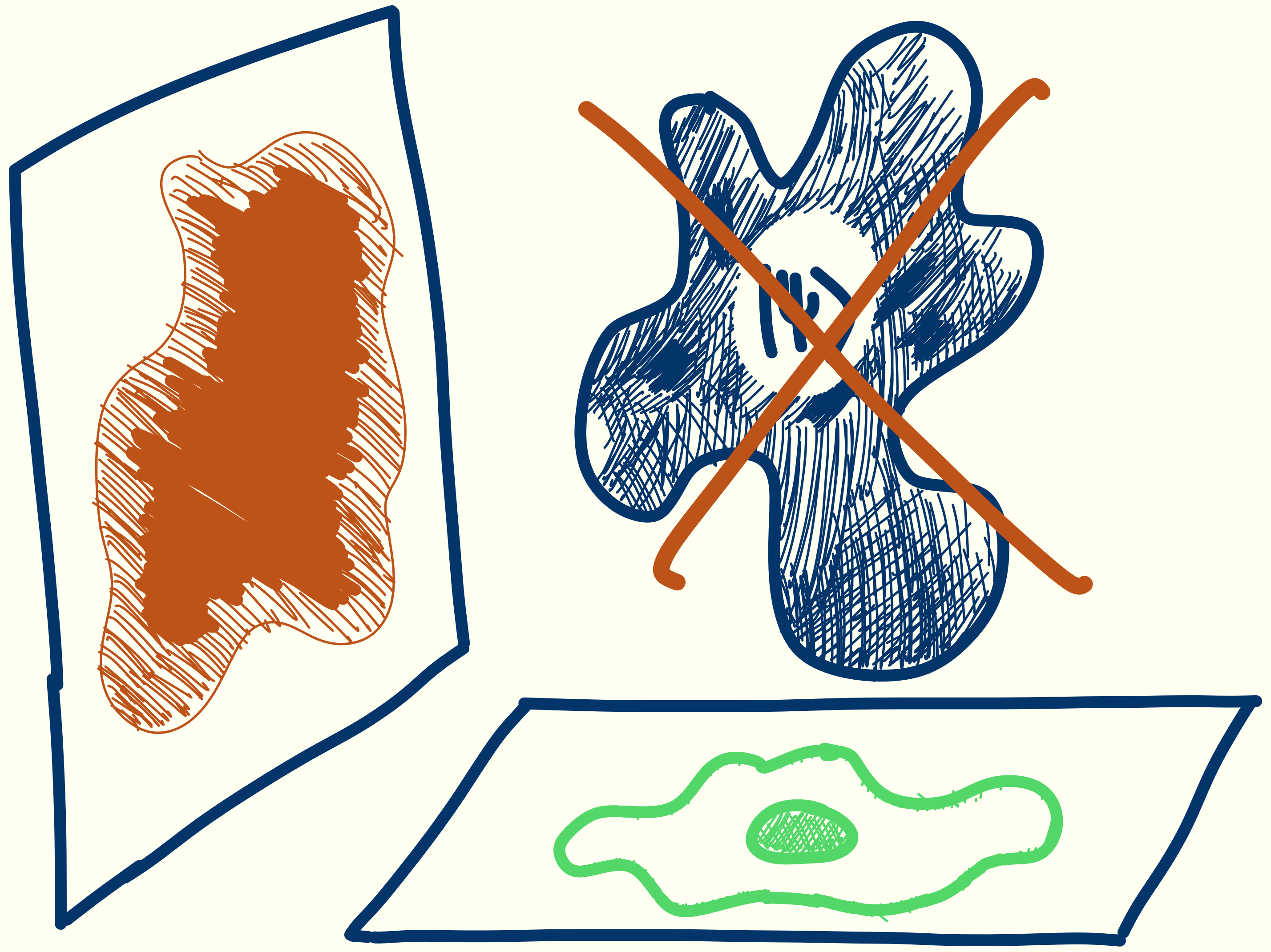


Strong yes instances

A pair (S, U) is a strong yes instance if:

- (S, U) is a yes instance of spectral Forrelation (i.e., $\geq 59/100$ spectrally Forrelated).
- For all $\Delta \subset S$ with $|\Delta| \leq \ell/100$, (Δ, U) is a no instance of spectral Forrelation (i.e., $\leq 57/100$ spectrally Forrelated).





Strong yes instances

Claim: (S, U) sampled from the Strong distribution will be a strong yes instance, except with probability $\ell 2^{n/6}$.

Strong yes instances

Claim: (S, U) sampled from the Strong distribution will be a strong yes instance, except with probability $\ell 2^{n/6}$.

Proof sketch: When we compute the expectation over U of the "Forrelation" matrix, we roughly get something that looks like

$$\mathbb{E}_U[\Pi_S \cdot H^{\otimes n} \cdot \Pi_U \cdot H^{\otimes n} \cdot \Pi_S] \approx \frac{1}{10} | S \rangle\langle S | + \frac{1}{2}\mathrm{id}$$

Strong yes instances

Claim: (S, U) sampled from the Strong distribution will be a strong yes instance, except with probability $\ell 2^{n/6}$.

Proof sketch: When we compute the expectation over U of the "Forrelation" matrix, we roughly get something that looks like

$$\mathbb{E}_U[\Pi_S \cdot H^{\otimes n} \cdot \Pi_U \cdot H^{\otimes n} \cdot \Pi_S] \approx \frac{1}{10} |S\rangle\langle S| + \frac{1}{2}\mathrm{id}$$

If this was really the matrix, then taking any $\Delta \times \Delta$ sub-matrix only get a small part of the mass of $|S\rangle$, making its operator norm close to 1/2.

Strong yes instances

Claim: (S, U) sampled from the Strong distribution will be a strong yes instance, except with probability $\ell 2^{n/6}$.

Proof sketch: When we compute the expectation over U of the "Forrelation" matrix, we roughly get something that looks like

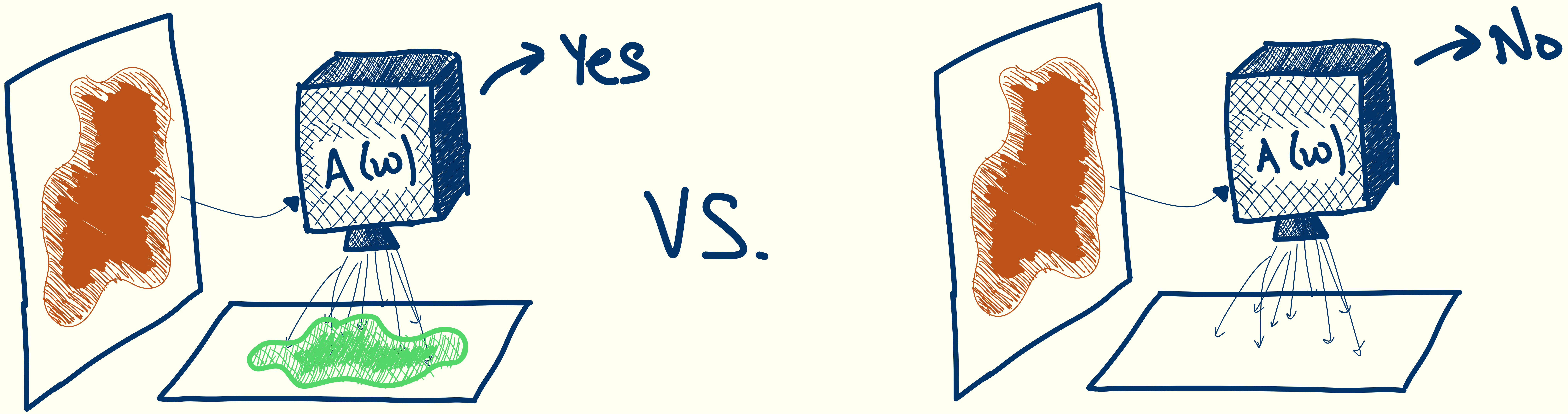
$$\mathbb{E}_U[\Pi_S \cdot H^{\otimes n} \cdot \Pi_U \cdot H^{\otimes n} \cdot \Pi_S] \approx \frac{1}{10} |S\rangle\langle S| + \frac{1}{2}\mathrm{id}$$

If this was really the matrix, then taking any $\Delta \times \Delta$ sub-matrix only get a small part of the mass of $|S\rangle$, making its operator norm close to $1/2$.

Using concentration bounds, we get that with very high probability, this happens.

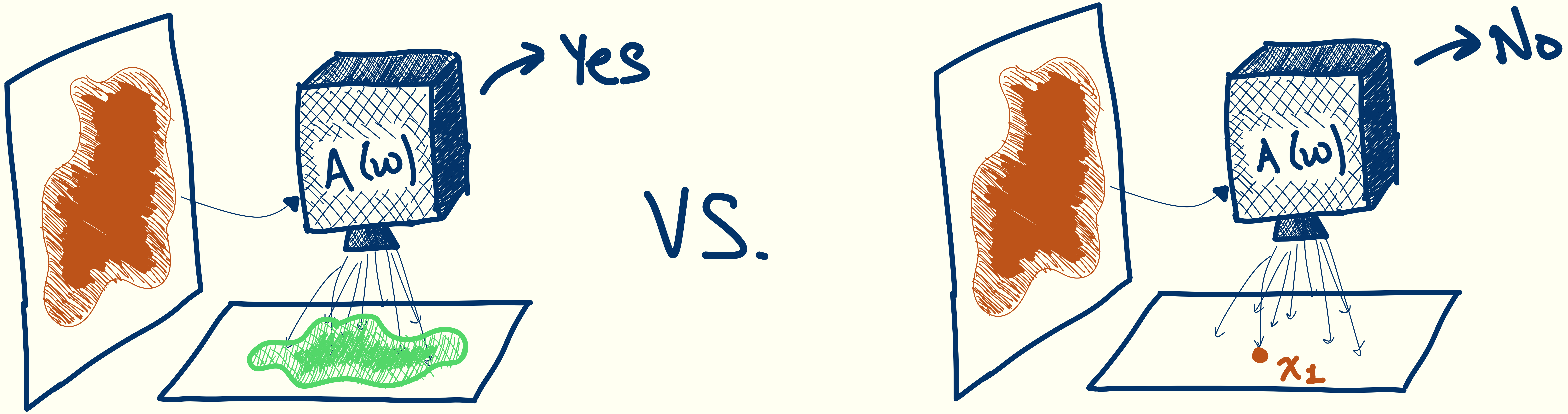
Strong yes instances can be sampled from

Any quantum query algorithm that distinguishes between (S, U) and $(\varnothing, U)$ must query a point in S pretty often ($\geq 1/3t$ chance per query), since otherwise the action of the oracles is identical.



Strong yes instances can be sampled from

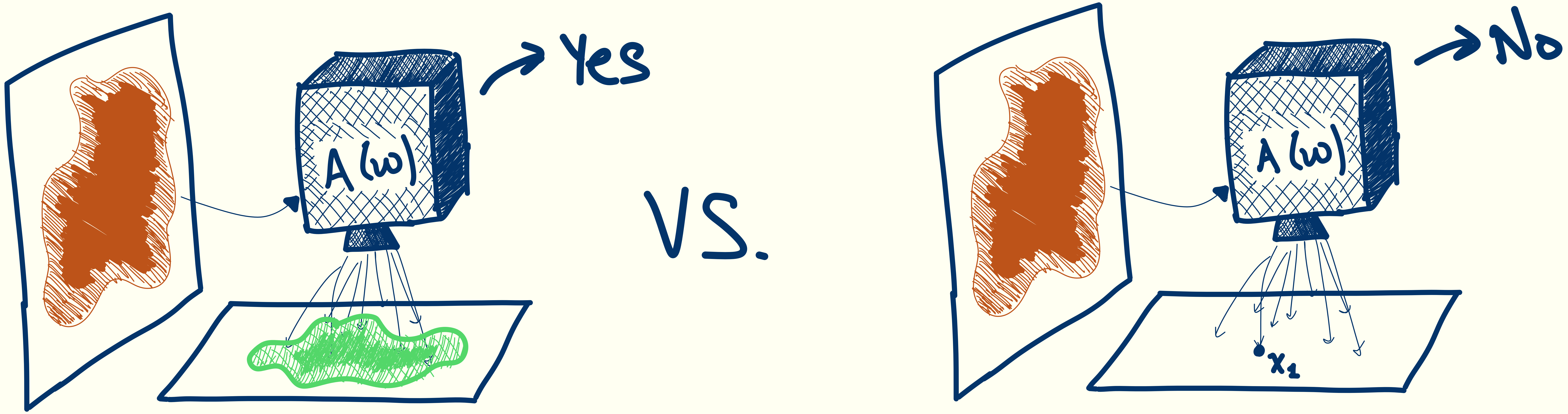
Any quantum query algorithm that distinguishes between (S, U) and $(\varnothing, U)$ must query a point in S pretty often ($\geq 1/3t$ chance per query), since otherwise the action of the oracles is identical.



Therefore, measuring a random query of the algorithm will yield a point in S with good probability, x_1.

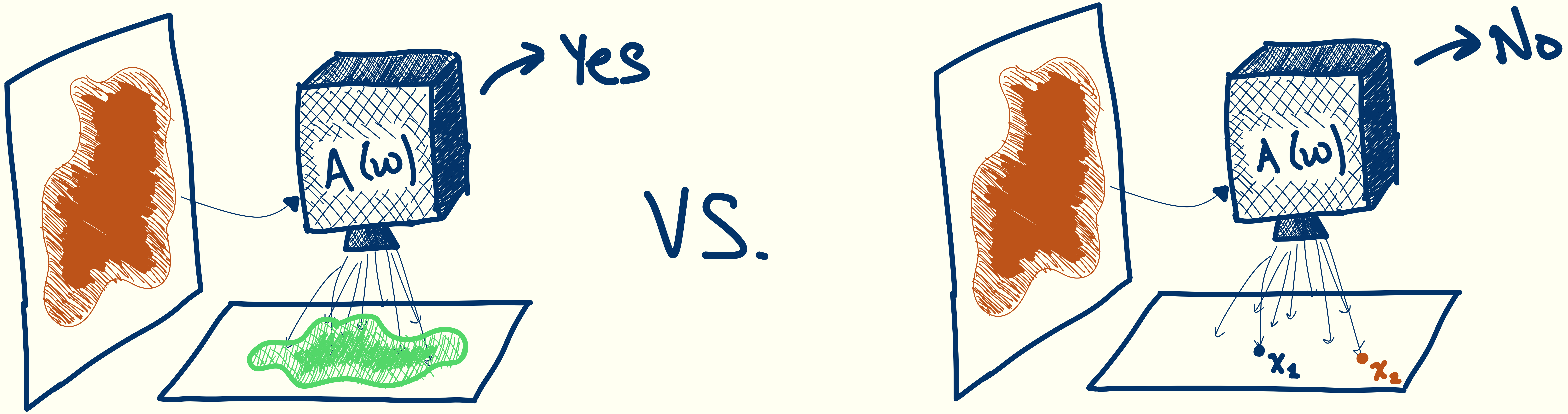
Strong yes instances can be sampled from

Any quantum query algorithm that distinguishes between (S, U) and $(\{x_1\}, U)$ must query a point in S pretty often ($\geq 1/3t$ chance per query), since otherwise the action of the oracles is identical.



Strong yes instances can be sampled from

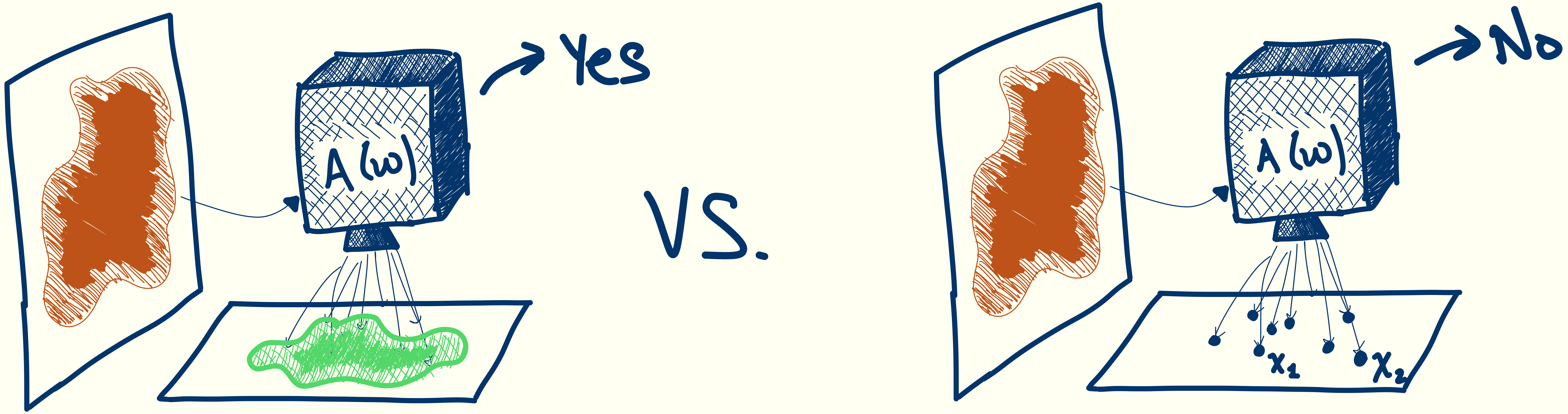
Any quantum query algorithm that distinguishes between (S, U) and $(\{x_1\}, U)$ must query a point in S pretty often ($\geq 1/3t$ chance per query), since otherwise the action of the oracles is identical.



Therefore, measuring a random query of the algorithm will yield a point in S with good probability, x_2.

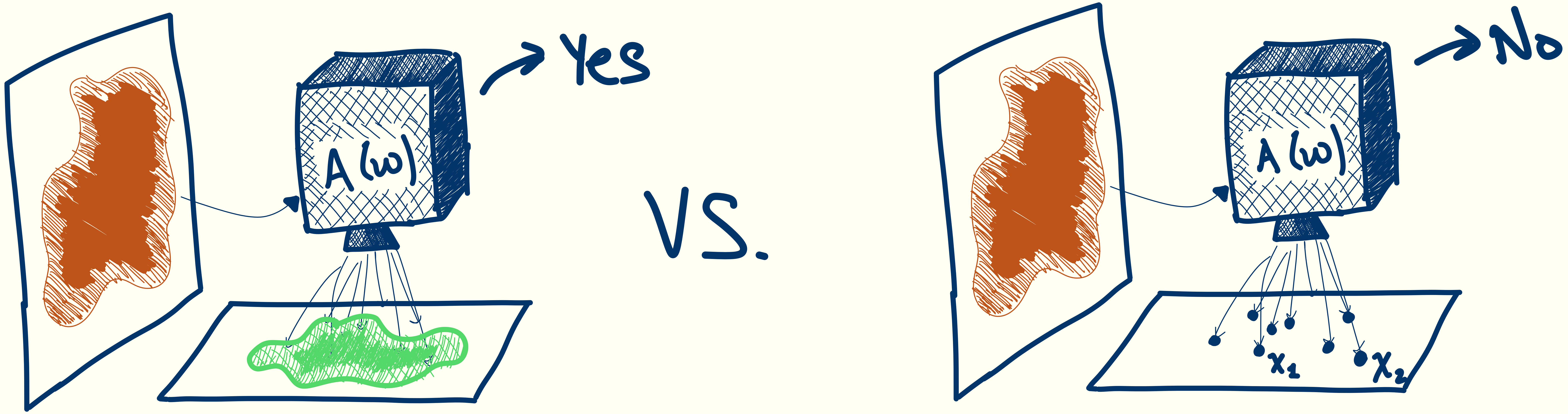
Strong yes instances can be sampled from

Because of the strong yes property, we can keep going until $\ell/100$ points have been sampled! This is the key step that uses the fact that the witness is classical.



Strong yes instances can be sampled from

Because of the strong yes property, we can keep going until $\ell/100$ points have been sampled! This is the key step that uses the fact that the witness is classical.



Given a QCMA algorithm, we can guess the classical witness and be correct with probability 2^{-q}.

Rest of the talk: sampling probability upper bound via the **compressed oracle technique**.

Recall: The theorem we're trying to prove:

Theorem 1: For all $v > 0$, and all quantum query algorithms making $T = T(n)$ queries to a set membership oracle for U, the probability, over Strong, that the algorithm outputs v distinct points from S is at most

$$\leq \left(\frac{\mathrm{poly}(v, T)}{\mathrm{poly}(2^n)}\right)^v.$$

Purification of quantum query algorithms

Say that you have an algorithm that acts on a distribution over oracles, i.e.,

$$\mathbb{E}_{\mathcal{O}\sim D}[A^{\mathcal{O}}]$$

Purification of quantum query algorithms

Say that you have an algorithm that acts on a distribution over oracles, i.e.,

$$\mathbb{E}_{\mathcal{O}\sim D}[A^{\mathcal{O}}]$$

We can imagine this came from tracing out part of the state:

$$\prod_{i=t}^{1}\left(\mathsf{pfO}\cdot A\right)|0\rangle\sum_{\mathcal{O}}\sqrt{D(\mathcal{O})}\,|\mathcal{O}\rangle$$

where $\mathsf{pfO} = \sum_{\mathcal{O},x}(-1)^{\mathcal{O}(x)}|x\rangle\langle x| \otimes |\mathcal{O}\rangle\langle\mathcal{O}|$

Purification of quantum query algorithms

Say that you have an algorithm that acts on a distribution over oracles, i.e.,

$$\mathbb{E}_{\mathcal{O}\sim D}[A^{\mathcal{O}}]$$

We can imagine this came from tracing out part of the state:

$$\prod_{i=t}^{1}\left(\mathsf{pfO}\cdot A\right)|0\rangle \sum_{\mathcal{O}} \sqrt{D(\mathcal{O})}\,|\mathcal{O}\rangle$$

where $\mathsf{pfO} = \sum_{\mathcal{O},x} (-1)^{\mathcal{O}(x)} |x\rangle\langle x| \otimes |\mathcal{O}\rangle\langle\mathcal{O}|$

The purifying register starts out unentangled and a successful algorithm will generate a specific type of entanglement (depending on the task)

Compressed oracles for Strong

Recall that we sampled Strong via the following:

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.

- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}\left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$

Compressed oracles for Strong

Recall that we sampled Strong via the following:

- We will first sample $\ell = 2^{n/10}$ many random elements $s_1, \ldots, s_\ell$. Let $|S\rangle$ be the uniform superposition over the points.
- We take U to be the heavy points of $H^{\otimes n}|S\rangle$, the Hadamard transform of $|S\rangle$:

$$\Pr[y \in U] = 1 - \frac{1}{2}\exp\left(-\frac{1}{10}\left|\langle y|H^{\otimes n}|S\rangle\right|^2\right)$$

Let's focus on purifying the distribution over S first!

Compressed oracles for Strong

The Fock basis is a way to write down a multi-set, similar to how we write subsets of $\{0,1\}^n$ as 2^n bit strings. Given a multi-set with ℓ_x copies of x, we associate it with a vector of 2^n non-negative integers:

$$|\ell_0, \ldots, \ell_{2^n-1}\rangle$$

Compressed oracles for Strong

The Fock basis is a way to write down a multi-set, similar to how we write subsets of $\{0,1\}^n$ as 2^n bit strings. Given a multi-set with ℓ_x copies of x, we associate it with a vector of 2^n non-negative integers:

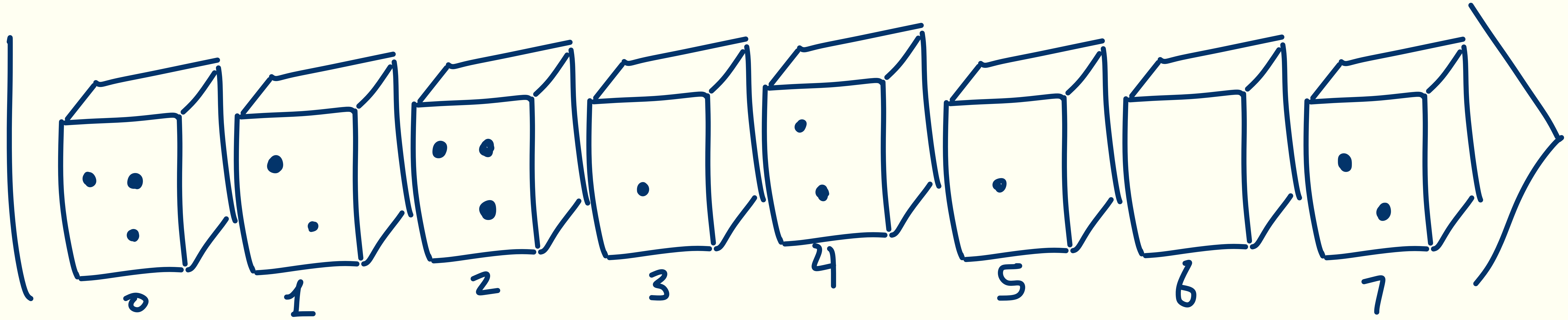
$$|\ell_0, \ldots, \ell_{2^n-1}\rangle$$

Then the uniform superposition over multi-sets is given by

$$\frac{1}{\sqrt{2^n}} \sum_{\vec{\ell}} \sqrt{\frac{\ell!}{\prod_x \ell_x!}} |\ell_0, \ldots, \ell_{2^n}\rangle .$$

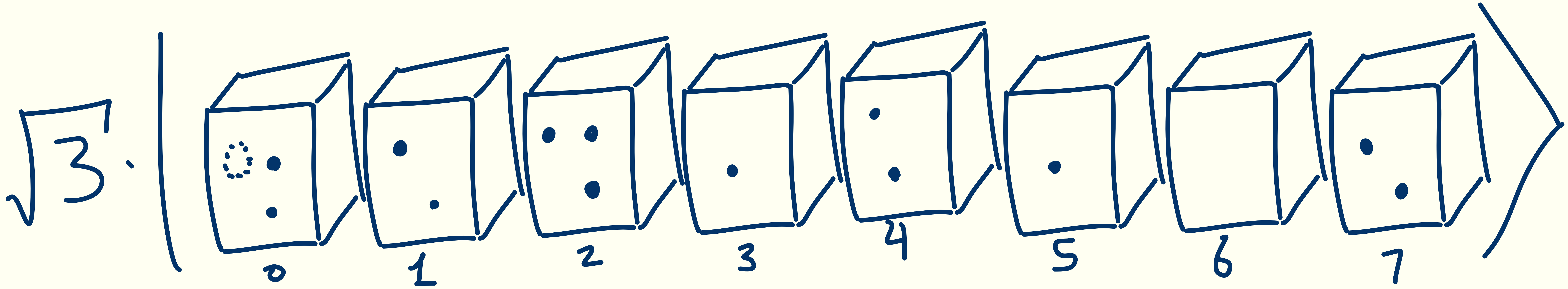
Compressed oracles for Strong: Bosons

Bosons are a mathematical representation of multi-sets used in physics. The "annihilation" and "creation" operators add and subtract elements (bosons).



Compressed oracles for Strong: Bosons

Bosons are a mathematical representation of multi-sets used in physics. The "annihilation" and "creation" operators add and subtract elements (bosons).

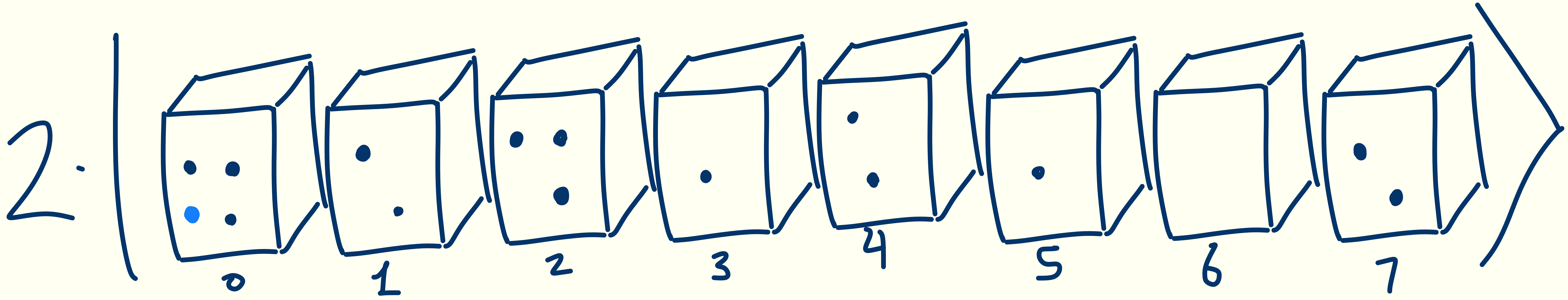


$$\hat{a}_x|\ell_0, \ldots, \ell_x, \ldots, \ell_{2^n-1}\rangle = \sqrt{\ell_x}|\ell_0, \ldots, \ell_x - 1, \ldots, \ell_{2^n-1}\rangle$$

(after $\hat{a}_0$)

Compressed oracles for Strong: Bosons

Bosons are a mathematical representation of multi-sets used in physics. The "annihilation" and "creation" operators add and subtract elements (bosons).



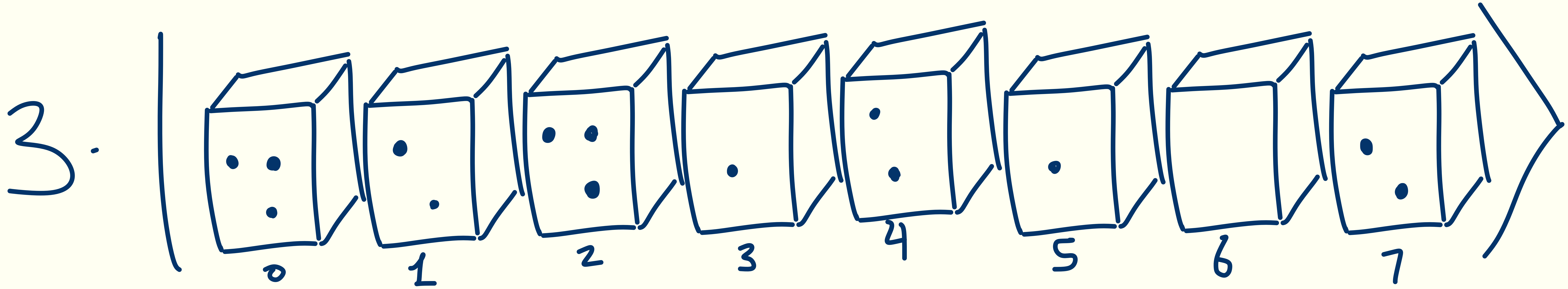
$$\hat{a}_x \,|\, \ell_0, \ldots, \ell_x, \ldots, \ell_{2^n-1}\rangle = \sqrt{\ell_x}\,|\,\ell_0, \ldots, \ell_x - 1, \ldots, \ell_{2^n-1}\rangle$$

(after $\hat{a}_0^\dagger$)

$$\hat{a}_x^\dagger \,|\, \ell_0, \ldots, \ell_x, \ldots, \ell_{2^n-1}\rangle = \sqrt{\ell_x + 1}\,|\,\ell_0, \ldots, \ell_x + 1, \ldots, \ell_{2^n-1}\rangle$$

Compressed oracles for Strong: Bosons

Bosons are a mathematical representation of multi-sets used in physics. The "annihilation" and "creation" operators add and subtract elements (bosons).



$$\hat{a}_x |\ell_0, \ldots, \ell_x, \ldots, \ell_{2^n-1}\rangle = \sqrt{\ell_x} |\ell_0, \ldots, \ell_x - 1, \ldots, \ell_{2^n-1}\rangle$$

$$\hat{a}_x^\dagger |\ell_0, \ldots, \ell_x, \ldots, \ell_{2^n-1}\rangle = \sqrt{\ell_x + 1} |\ell_0, \ldots, \ell_x + 1, \ldots, \ell_{2^n-1}\rangle$$

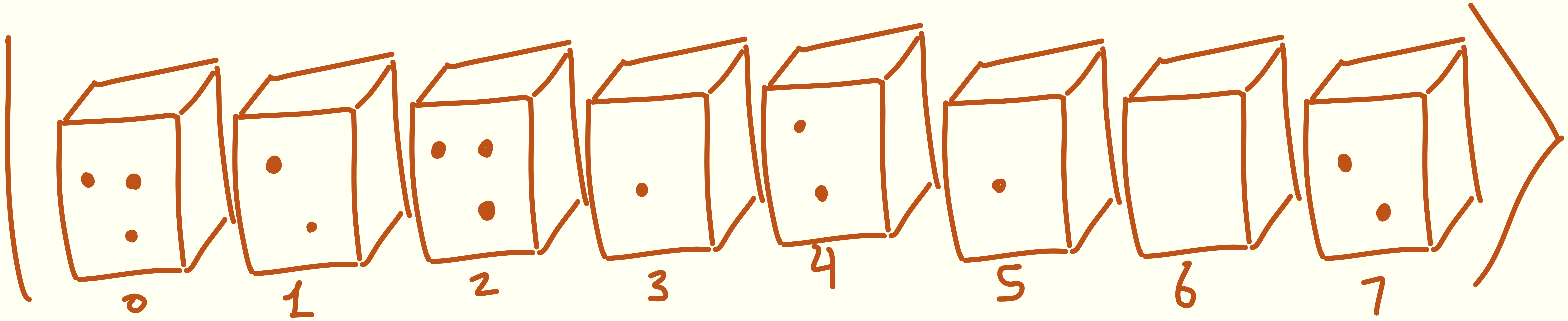
(after $\hat{n}_0$)

Taking the product, the number operator $\hat{n}_x = \hat{a}_x^\dagger \hat{a}_x$ is diagonal in the position Fock basis, and applies a scaling of the number of bosons in the x'th mode.

Compressed oracles for Strong: Bosons

We can also define a "Hadamard" basis for the bosons, with the analogous operators:

$$\widetilde{a}_y = \frac{1}{\sqrt{2^n}} \sum_x (-1)^{y \cdot x} \hat{a}_x \quad \text{and} \quad \widetilde{a}_y^\dagger = \frac{1}{\sqrt{2^n}} \sum_x (-1)^{y \cdot x} \hat{a}_x^\dagger$$

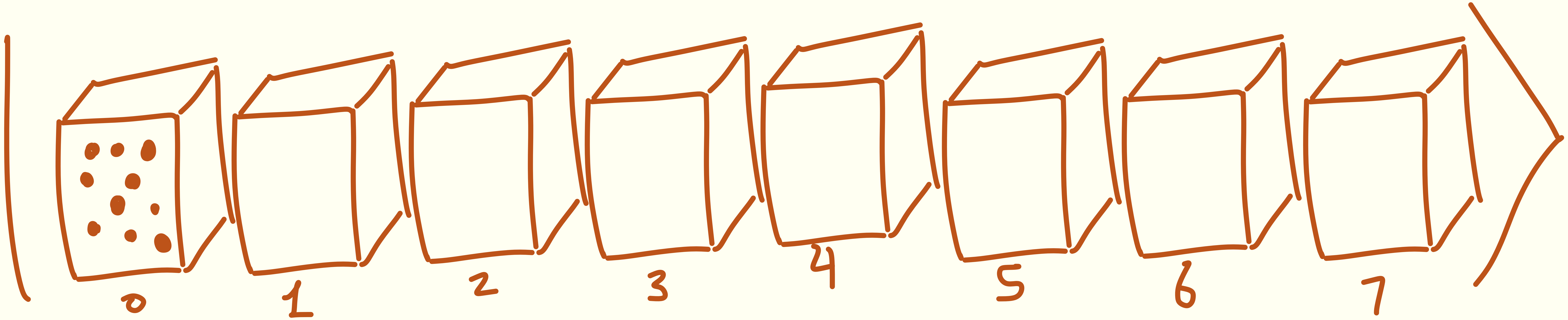


As a matter of notation, we also define $|\text{vac}\rangle$ to be the state $|0,\ldots,0\rangle$.

Compressed oracles for Strong: Bosons

Claim: The purification of a random multi-set is:

$$|\text{init}\rangle = \frac{1}{\sqrt{\ell!}}\left(\tilde{a}_0^\dagger\right)^\ell |\text{vac}\rangle$$



Compressed oracles for Strong: Bosons

Claim: The purification of a random multi-set is:

$$|\text{init}\rangle = \frac{1}{\sqrt{\ell!}} \left(\tilde{a}_0^\dagger\right)^\ell |\text{vac}\rangle$$

Proof: Expand out the expression:

$$\frac{1}{\sqrt{\ell!}} \left(\tilde{a}_0^\dagger\right)^\ell |\text{vac}\rangle = \frac{1}{\sqrt{2^{n\ell} \cdot \ell!}} \sum_{s_1,\ldots,s_\ell} \hat{a}_{s_1}^\dagger \ldots \hat{a}_{s_\ell}^\dagger |\text{vac}\rangle$$

If you deal with the coefficients (and multiplicities of the multi-sets), it works out. 😊

Compressed oracles for Strong: Action of U

Roughly, querying U at some fixed y is like querying the squared Fourier coefficient $\gamma_y^{(S)}$.
What happens when we apply the diagonal matrix

$$\sum_S \gamma_y^{(S)} |\mathrm{Fock}(S)\rangle\langle\mathrm{Fock}(S)|\ ?$$

Compressed oracles for Strong: Action of U

Roughly, querying U at some fixed y is like querying the squared Fourier coefficient $\gamma_y^{(S)}$.
What happens when we apply the diagonal matrix

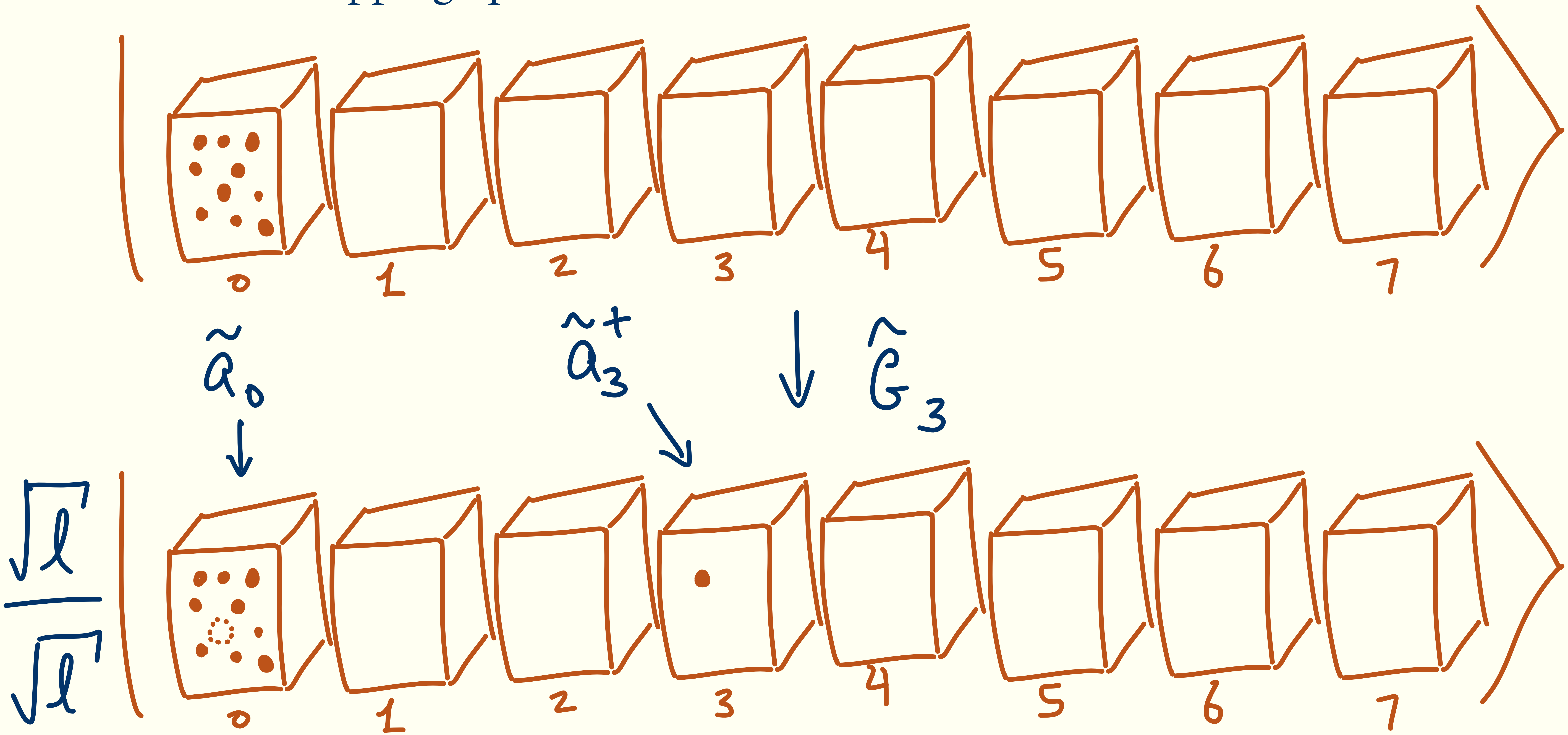
$$\sum_S \gamma_y^{(S)} \,|\,\mathrm{Fock}(S)\rangle\langle\mathrm{Fock}(S)\,|\ ?$$

Let's define the momentum hopping and double hopping operators

$$\widetilde{G}_y = \frac{1}{\sqrt{\ell}} \sum_{x\in\{0,1\}^n} \widetilde{a}^{\dagger}_{x\oplus y}\widetilde{a}_x \quad \text{and} \quad \widetilde{H}_y = \frac{1}{\ell} \sum_{x,x'\in\{0,1\}^n} \widetilde{a}^{\dagger}_{x\oplus y}\widetilde{a}^{\dagger}_{x'\oplus y}\widetilde{a}_x\widetilde{a}_{x'}$$

Compressed oracles for Strong: Action of U

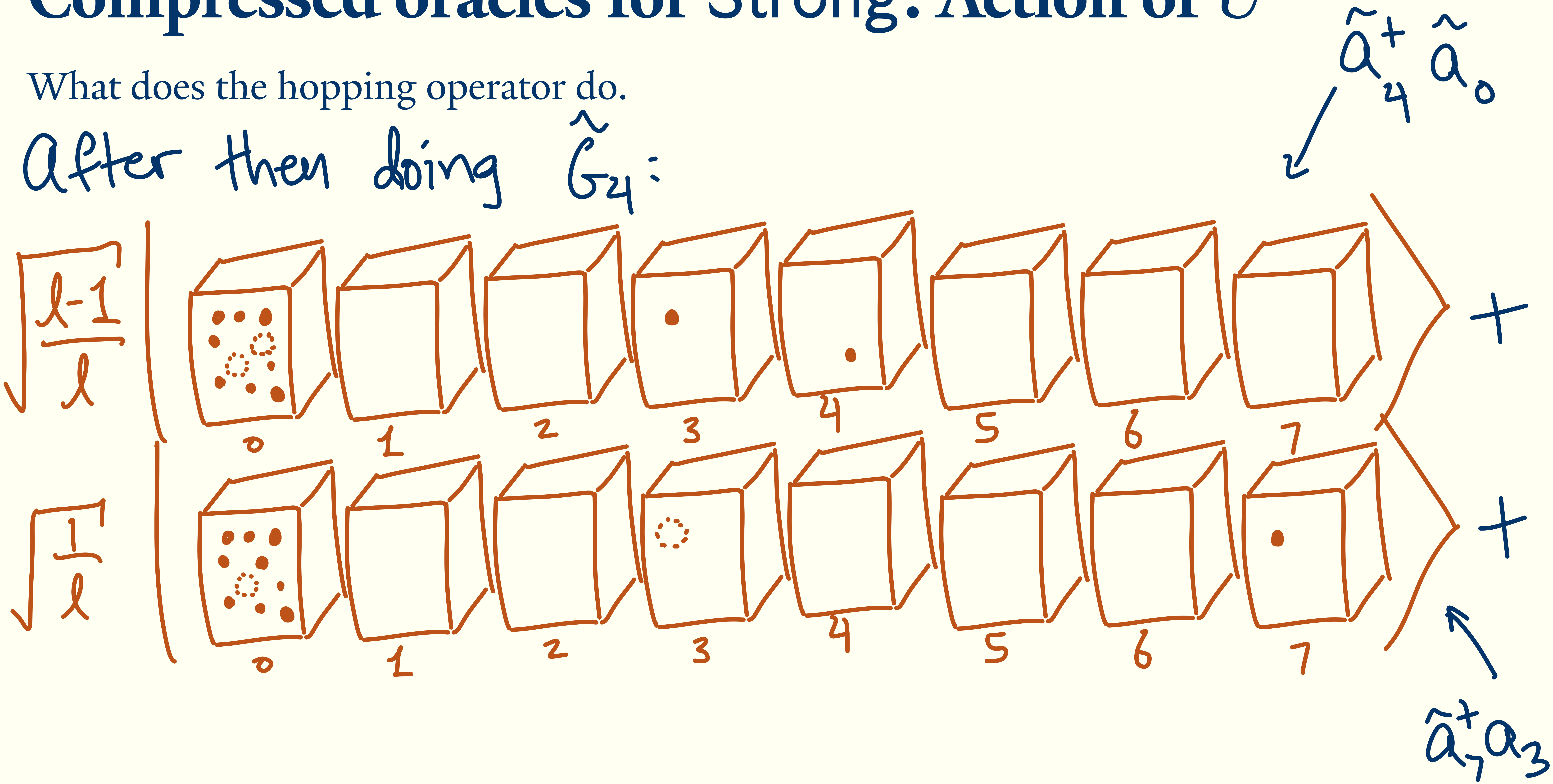
What does the hopping operator do.



Compressed oracles for Strong: Action of U

What does the hopping operator do.

After then doing $\tilde{G}_4$:



Compressed oracles for Strong: Action of U

Claim: The diagonal matrix that applies the squared Fourier coefficient is actually:

$$\sum_S \gamma_y^{(S)} |\mathrm{Fock}(S)\rangle\langle\mathrm{Fock}(S)| = \widetilde{H}_y + \mathrm{id}$$

Compressed oracles for Strong**: Action of** U

Claim: The diagonal matrix that applies the squared Fourier coefficient is actually:

$$\sum_S \gamma_y^{(S)} |\mathrm{Fock}(S)\rangle\langle\mathrm{Fock}(S)| = \widetilde{H}_y + \mathrm{id}$$

Proof: We can expand out a position Fock state in the momentum basis and directly compute the action of the hopping operator (the double hopping is the square):

$$\widetilde{G}_y \hat{a}^\dagger_{s_1} \ldots \hat{a}^\dagger_{s_\ell} |\mathrm{vac}\rangle = \widetilde{G}_y \sum_{t_1,\ldots,t_\ell} \left(\prod_i (-1)^{t_i \cdot s_i} \right) \widetilde{a}^\dagger_{t_1} \ldots \widetilde{a}^\dagger_{t_\ell} |\mathrm{vac}\rangle$$

When we apply the hop and re-index the sum, we see that we just get a phase kickback!

Compressed oracles for Strong: Summary

So far, what we (hopefully) learned:

- In momentum space, the initial state of the S register looks like ℓ bosons in the 0 -momentum mode.

Compressed oracles for Strong: Summary

So far, what we (hopefully) learned:

- In momentum space, the initial state of the S register looks like ℓ bosons in the 0 -momentum mode.

- Applying a query to U roughly looks like applying the double hopping operator $\widetilde{H}_y + \mathrm{id}$.

Compressed oracles for Strong: Summary

So far, what we (hopefully) learned:

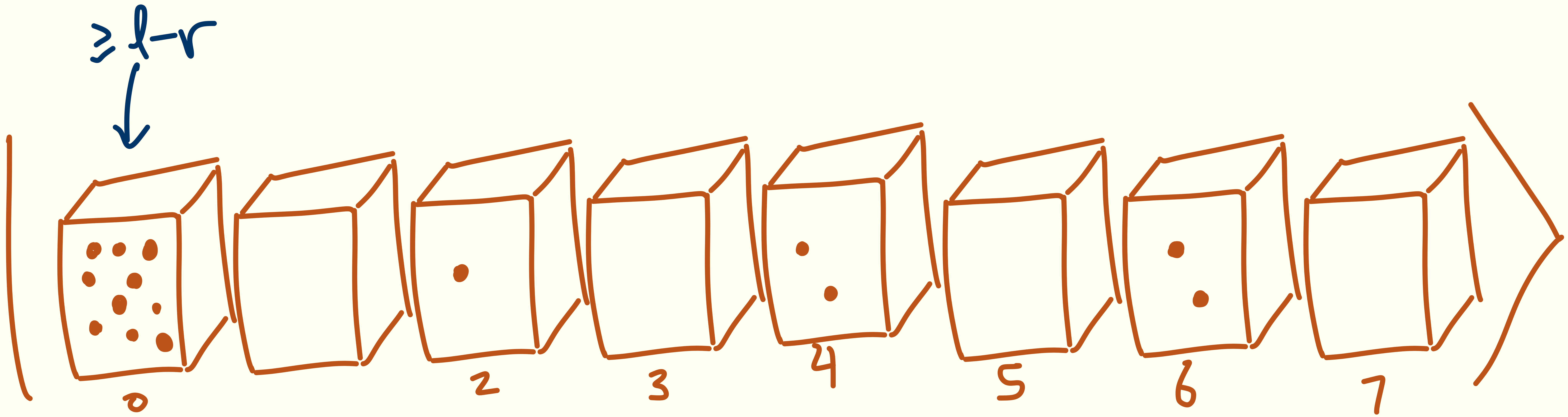
- In momentum space, the initial state of the S register looks like ℓ bosons in the 0-momentum mode.
- Applying a query to U roughly looks like applying the double hopping operator $\widetilde{H}_y + \mathrm{id}$.

How does this let us prove a sampling probability upper bound?

Quasi-even condensates

A (r, o)-quasi-even condensate is a momentum Fock state $|\ell_0, \ldots, \ell_{2^n}\rangle$ that satisfies:

Condensate: $\ell_0 \geq \ell - r$, i.e., almost all of the bosons are in their initial position.

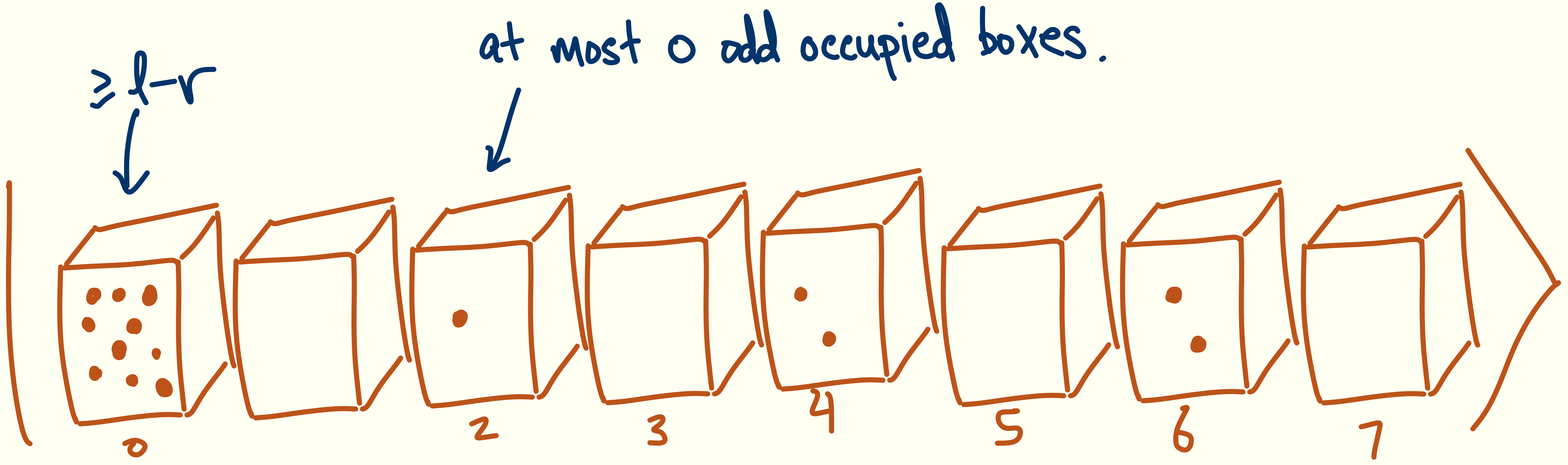


Quasi-even condensates

A (r, o)-quasi-even condensate is a momentum Fock state $|\ell_0, \ldots, \ell_{2^n}\rangle$ that satisfies:

Condensate: $\ell_0 \geq \ell - r$, i.e., almost all of the bosons are in their initial position.

Quasi-even: At most o of the non-zero indices are odd.



Sampling bounds on quasi-even condensates

Claim: Let $|\psi\rangle$ be a state that is supported entirely on (r, o)-quasi-even condensate, then the following bound holds for all collections $z_1, \ldots, z_v \in \{0,1\}^n$:

$$\langle\psi|n_{z_1}\ldots, n_{z_\ell}|\psi\rangle \leq \left(\text{poly}(v, r) \cdot \frac{\sqrt{\ell}}{2^{n/4}}\right)^v$$

Sampling bounds on quasi-even condensates

Claim: Let $|\psi\rangle$ be a state that is supported entirely on (r, o)-quasi-even condensate, then the following bound holds for all collections $z_1, \ldots, z_v \in \{0,1\}^n$:

$$\langle\psi| n_{z_1} \ldots, n_{z_\ell} |\psi\rangle \leq \left(\mathrm{poly}(v, r) \cdot \frac{\sqrt{\ell}}{2^{n/4}} \right)^v$$

This number upper bounds the sampling success probability (applying Markov's inequality). If we knew that the algorithm's purified state was supported only on quasi-even condensates, we would be done.

Sampling bounds on quasi-even condensates

Intuition for why quasi-evenness is the right notion:

- Imagine the position shift operator $\mathrm{Shift}_x^\dagger \cdot a_y^\dagger \cdot \mathrm{Shift}_x = a_{x \oplus y}^\dagger$.

Sampling bounds on quasi-even condensates

Intuition for why quasi-evenness is the right notion:

- Imagine the position shift operator $\mathrm{Shift}_x^\dagger \cdot a_y^\dagger \cdot \mathrm{Shift}_x = a_{x \oplus y}^\dagger$.
- If we apply it to a momentum operator, we get $\mathrm{Shift}_x^\dagger \cdot \widetilde{a}_y^\dagger \cdot \mathrm{Shift}_x = (-1)^{x \cdot y} \widetilde{a}_y^\dagger$, meaning

Sampling bounds on quasi-even condensates

Intuition for why quasi-evenness is the right notion:

- Imagine the position shift operator $\mathrm{Shift}_x^\dagger \cdot a_y^\dagger \cdot \mathrm{Shift}_x = a_{x\oplus y}^\dagger$.
- If we apply it to a momentum operator, we get $\mathrm{Shift}_x^\dagger \cdot \widetilde{a}_y^\dagger \cdot \mathrm{Shift}_x = (-1)^{x\cdot y}\widetilde{a}_y^\dagger$, meaning

$$\mathrm{Shift}_x^\dagger \cdot \widetilde{a}_0^\dagger \cdot \mathrm{Shift}_x = \widetilde{a}_0^\dagger \text{, and}$$
$$\mathrm{Shift}_x^\dagger \cdot \left(\widetilde{a}_y^\dagger\right)^2 \cdot \mathrm{Shift}_x = \mathrm{Shift}_x^\dagger \cdot \widetilde{a}_y^\dagger \cdot \mathrm{Shift}_x \cdot \mathrm{Shift}_x^\dagger \cdot \left(\widetilde{a}_y^\dagger\right)^2 \cdot \mathrm{Shift}_x = \left(\widetilde{a}_y^\dagger\right)^2$$

Sampling bounds on quasi-even condensates

Intuition for why quasi-evenness is the right notion:

- Imagine the position shift operator $\mathrm{Shift}_x^\dagger \cdot a_y^\dagger \cdot \mathrm{Shift}_x = a_{x\oplus y}^\dagger$.
- If we apply it to a momentum operator, we get $\mathrm{Shift}_x^\dagger \cdot \widetilde{a}_y^\dagger \cdot \mathrm{Shift}_x = (-1)^{x\cdot y}\widetilde{a}_y^\dagger$, meaning

  $$\mathrm{Shift}_x^\dagger \cdot \widetilde{a}_0^\dagger \cdot \mathrm{Shift}_x = \widetilde{a}_0^\dagger \text{, and}$$
  $$\mathrm{Shift}_x^\dagger \cdot \left(\widetilde{a}_y^\dagger\right)^2 \cdot \mathrm{Shift}_x = \mathrm{Shift}_x^\dagger \cdot \widetilde{a}_y^\dagger \cdot \mathrm{Shift}_x \cdot \mathrm{Shift}_x^\dagger \cdot \left(\widetilde{a}_y^\dagger\right)^2 \cdot \mathrm{Shift}_x = \left(\widetilde{a}_y^\dagger\right)^2$$

- This means that bosons in the condensate (0-momentum) and paired up are spread out uniformly among the positions (relative to the adversary's state), and therefore hard to guess.

Sampling bounds on quasi-even condensates

The final step is to show that an adversary querying the purified U is supported mostly on quasi-even condensates.

Roughly: The double hopping operator picks random bosons, so as long as it touches a r-condensate, most of its “weight” is two bosons from 0 to y momentum (on query y).

Sampling bounds on quasi-even condensates

The final step is to show that an adversary querying the purified U is supported mostly on quasi-even condensates.

Roughly: The double hopping operator picks random bosons, so as long as it touches a r-condensate, most of its "weight" is two bosons from 0 to y momentum (on query y).

After T-queries, we would expect to have a $\sim 2T$-condensate, and fewer than $v/4$ unpaired bosons, except with probability roughly $\left(vT^3\sqrt{\ell}/2^{n/4}\right)^v$.

Missing details

Many details were omitted, happy to explain them in more detail after these slides!

Missing details

Many details were omitted, happy to explain them in more detail after these slides!

- Queries to U are actually more like measurements of Kraus operators $E_0 = (1 - \exp(-\widetilde{G}_y^2/10))$ and $E_1 = \sqrt{\exp(-\tilde{G}_y^2/10)(2 - \exp(-\tilde{G}_y^2/10))}$.

Missing details

Many details were omitted, happy to explain them in more detail after these slides!

- Queries to U are actually more like measurements of Kraus operators $E_0 = (1 - \exp(-\widetilde{G}_y^2/10))$ and $E_1 = \sqrt{\exp(-\tilde{G}_y^2/10)(2 - \exp(-\tilde{G}_y^2/10))}$.
- Applying a flat polynomial approximation, we can show that these are close to polynomial in $\widetilde{G}_y^2$ whose degree is $\sim T^{10}$, which then gives us a condensate.

Missing details

Many details were omitted, happy to explain them in more detail after these slides!

- Queries to U are actually more like measurements of Kraus operators $E_0 = (1 - \exp(-\widetilde{G}_y^2/10))$ and $E_1 = \sqrt{\exp(-\tilde{G}_y^2/10)(2 - \exp(-\tilde{G}_y^2/10))}$.
- Applying a flat polynomial approximation, we can show that these are close to polynomial in $\widetilde{G}_y^2$ whose degree is $\sim T^{10}$, which then gives us a condensate.
- But, this polynomial is not necessarily bounded anymore, so we need to bring in tools from perturbation theory (the Dyson series) to prove the quasi-even property.

Main theorems

Theorem 1: For all $v > 0$, and all quantum query algorithms making $T = T(n)$ queries to a set membership oracle for U, the probability, over Strong, that the algorithm outputs v distinct points from S is at most

$$\leq \left(\frac{\mathrm{poly}(v, T)}{\mathrm{poly}(2^n)}\right)^v.$$

Theorem 2: If there exists a QCMA algorithm, making $t = t(n)$ queries to (S, U) and taking a witness of length $q = q(n)$, then for all $0 < v < \ell/100$, there is a query algorithm making vt queries to U that outputs v distinct points from S with probability

$$\geq 2^{-q}\left(\frac{1}{36t^2}\right)^v$$

Main theorems

Theorem 1: For all $v > 0$, and all quantum query algorithms making $T = T(n)$ queries to a set membership oracle for U, the probability, over Strong, that the algorithm outputs v distinct points from S is at most

$$\leq \left(\frac{\mathrm{poly}(v, T)}{\mathrm{poly}(2^n)}\right)^v.$$

Theorem 2: If there exists a QCMA algorithm, making $t = t(n)$ queries to (S, U) and taking a witness of length $q = q(n)$, then for all $0 < v < \ell/100$, there is a query algorithm making vt queries to U that outputs v distinct points from S with probability

$$\geq 2^{-q}\left(\frac{1}{36t^2}\right)^v$$

When $v \sim 1000q$, we get a contradiction

Takeaways

Takeaways

- **Quantum proofs are really powerful!**
  → That power is what we think makes them not reusable!
  → Our proof finds a task (sampling) that should be really hard, and shows that a reusable proof would be too good to be true.

Takeaways

- **Quantum proofs are really powerful!**
  → That power is what we think makes them not reusable!
  → Our proof finds a task (sampling) that should be really hard, and shows that a reusable proof would be too good to be true.
- **Small structural changes can have a huge impact!**
  → The bosonic compressed oracle came from not requiring that S has exactly ℓ elements, and allowing it to be a multi-set with independent elements instead.
  → This removal of structure allowed us to understand queries to the Fourier transform of an oracle way better than we could before!

Takeaways

- **Quantum proofs are really powerful!**
  → That power is what we think makes them not reusable!
  → Our proof finds a task (sampling) that should be really hard, and shows that a reusable proof would be too good to be true.
- **Small structural changes can have a huge impact!**
  → The bosonic compressed oracle came from not requiring that S has exactly ℓ elements, and allowing it to be a multi-set with independent elements instead.
  → This removal of structure allowed us to understand queries to the Fourier transform of an oracle way better than we could before!
- **Much more work is needed!**
  → Understanding oracles with structure seems to require an understanding that structure, seem to be annoying to deal with using general methods.
  → To understand other oracles (expander mixing problem, Yamakawa-Zhandry, etc.), we will need more specific tools, or a big leap in understanding of quantum algorithms.

Open questions

Open questions

- **Can we find new constructions/security proofs for quantum money?**
  → Our ideas lie in the intersection of ideas used for quantum money (subset states ↔ subspace states, Fourier transform of S ↔ Fourier transform for group actions).
  → We also prove a separation between UnclonableQMA and QMA, feels like we should be able to say something about quantum money, but what?

Open questions

- **Can we find new constructions/security proofs for quantum money?**
  → Our ideas lie in the intersection of ideas used for quantum money (subset states ↔ subspace states, Fourier transform of S ↔ Fourier transform for group actions).
  → We also prove a separation between UnclonableQMA and QMA, feels like we should be able to say something about quantum money, but what?

- **Can we use our oracle/techniques to solve other problems in query complexity?**
  → BQP/qpoly versus BQP/poly?
  → QMA search-to-decision?

Open questions

- **Can we find new constructions/security proofs for quantum money?**
  → Our ideas lie in the intersection of ideas used for quantum money (subset states ↔ subspace states, Fourier transform of S ↔ Fourier transform for group actions).
  → We also prove a separation between UnclonableQMA and QMA, feels like we should be able to say something about quantum money, but what?

- **Can we use our oracle/techniques to solve other problems in query complexity?**
  → BQP/qpoly versus BQP/poly?
  → QMA search-to-decision?

- **Is there a connection to the Aaronson-Ambainis conjecture?**
  → Both Liu-Mutreja-Yuen'24 and Zhandry'24 showed that there is a connection between QCMA versus QMA and pseudorandomness against quantum algorithms.
  → Our proof didn't say anything about this, but could you use our techniques?

Thanks for listening!