Greg Kuperberg's Lectures on

Introduction to Quantum Computation

Sanchayan Dutta
dutta@ucdavis.edu

Contents

1 Lecture 1 (12 November 2021)

1.1 Fundamental Goal

To characterize the category of realistic maps between states.

We have two conclusions:

1. If a map $E\colon A^{\#} \to B^{\#}$ is realistic then it is TPCP. If E is not linear it violates classical superposition. If it's not CP then it and together with a companion thing can create negative probabilities or non-real probabilities. If it is not TP is doesn't preserve probability.

2. (**Stinespring-type theorem**) If E is TPCP, then you can produce it as a composition of realistic components. In fact not very many of them – just a factorization consisting of ancilla pure states on a Hilbert, followed by the transpose of an algebra homomorphism.

Say that $A = M(a)$ and $B = M(b)$ then (2) is equivalent to a structure theorem due to Krauss for the structure of E. At a conceptual level this structure theorem plays an important role and in this way of doing things quantum probability (and quantum superposition) arises as a corollary of classical probability (and classical superposition).

Kraus' theorem (named after Karl Krauss) characterizes CP maps that model quantum operations between quantum states. Informally, the theorem ensures that the action of any such quantum operation E on a state ρ can always be written as $E(\rho) = \sum x\rho x^*$ for some set of operators $\{x_k\}_k$ satisfying $\sum_k x_k^* x_k = \mathbf{1}$ where $\mathbf{1}$ is the identity operator.

$E(\rho) = \sum x\rho x^*$ is a classical superposition of terms. If ρ is a pure state of the form $|\psi\rangle\langle\psi|$ then $x|\psi\rangle\langle\psi|x^*$ is a linear operator on the pure state. The $x\rho x^*$ part is a quantum superposition.

Note: Here in $E\colon A^{\#} \to B^{\#}$, A and B need not necessarily be qudits. They can be semi-quantum in various ways.

In this way, we get our category QProb and the finite-dimensional part qProb.

1.2 QProb and qProb

Theorem: If E (in QProb) is *reversible*, E^{-1} exists and is also TPCP, and E comes from an algebra isomorphism. This means there's a restricted way to go backwards. The only way this can happen is if all of the algebra apparatus is preserved for Alice and Bob.

(In terms of category theory, if you have any category with a set of reversible maps, the maps are said to have inverses in that category.)

Theorem: Any algebra isomorphism between qudits or $\mathcal{L}(H_A)$ and $\mathcal{L}(H_B)$ is given by a unitary operator. This is another example of getting quantum superposition and quantum linearity out of classical superposition and classical linearity.

(In pure math, if C is a category and $f\colon A \to B$ in the category C has an inverse f^{-1} also in the same category, then f is called invertible. Well, f is also called an isomorphism. In physics one says, f is reversible. E may represent the evolution of system. We may ask when the time reversal is well founded under the laws of physics.)

An example of TPCPs: Unitary operators and isomorphisms and more generally, automorphisms.

Every von Neumann algebra has unitary elements. In any VNA M, there is a unitary subgroup $U(M)$ consisting of the solutions to $u^*u = uu^* = 1$. If $M = \mathcal{L}(H)$ then the corresponding algebra automorphism is $x \to uxu^*$. Physicists and QIT folks will call this transformation unitary whereas operator algebraists will call this kind of automorphism inner.

- $U(M) = M(S^1)$ which are the circle-valued observables
- $U(M)$ is friends with $M_{\mathbb{R}}$, $M_{\mathbb{Z}/2\mathbb{Z}}$, etc.

• $M_{\text{normal}} = \{z \text{ s.t. } zz^* = z^*z\}$ = one-shot measurements. The order in which you measure the real and imaginary parts of z does not matter since they commute with each other. See: physics.stackexchange.com/a/82616 for more details. Unitary is a special case of normal.
• $U(M)$ represents measurements that take values in a unit circle on the complex plane.
• If M is commutative, the only automorphisms $x \to uxu*$ is the identity.

1.3 Modelling Measurements using TPCPs

(1) **Automorphisms and isomorphisms**
(2) **Visible measurement**

M = any system A = a classical observer, for starters a bit

Given a homomorphism f from A to M there is then a measurement TPCP.

$E\colon M^\# \to (M \otimes A)^\#$ where afterwards M holds the posterior state and A holds the measured value. We can suppose that A injects into M. So we need a algebra homomorphism and we can suppose that it is injective. If it's not injective then the measurment will have redundancies. In some way or another you will be measuring the same thing multiple times.

If A is a finite digit with a configurations, you can take E apart into a disjoint booleans $b_1, \ldots, b_a$. Disjoint means $b_j b_k = 0$ for $j \neq k$. So they are simultaneously measurable. Then the posterior state in M is a sum over the posterior state formula from before:

$$E(\rho) = \sum b_a \rho b_a \otimes [a]$$

where $[a]$ is the output value. This is the qudit case for M and $b_k = f(k)$.

(3) **Discard or Departure**

We have the final system $\mathbb{C} = M(1)$ and for any other A, there is a unique TPCP $A^\# \to \mathbb{C}^\#$ interpreted as discarding A or A leaves the room.

Final and Initial Objects

If C is a category, a final object is one that has exactly one morphism from any other object. An initial object has exactly one morphism to any other object. All final objects are isomorphic. Also all initial objects. Suppose A and B are two final objects. Then there can be only one map from A to B and only map from B to A.

Example: The category Set has both a final object and an initial object and they aren't isomorphic to each other.

Think of a set such that there is always only one function from it. The empty set is initial. Think of a set such there is only one function to it. 1-point sets are final.

Example: The category Vect. Think of a vector space such that there is only one linear map from it. The 0-dimensional vector space $\{0\}$. Now think of a vector space such that there is only one linear map to it. The 0-dimensional vector space again! So this vector space is both initial and final.

So QProb has a final object. The VNA $M = \mathbb{C}$ representing the complex numbers. QProb doesn't have an initial object. Classical probability too doesn't have an initial object but it does have a final object. Which means there is only one way to "destroy" but many ways to "create". This is because (say) you can create a qudit in any case, but there is only one way to destroy/discard a qudit.

In fact, the maps from $\mathbb{C}^\# \to A^\#$ are a copy of A^Δ. You can reproduce the states on A as TPCPs from non-existence to A.

Measurement Process

Combining (2) and (3), measurement into A followed by discarding A is the hidden measurement TPCP. So Alice can come in, measure (using a homorphism) and leave. She will never tell you what was measurement. If the system M was non-commutative, it's state would have changed.

If M is a qudit and A is finite, $\rho \to \sum b_k \rho b_k$.

E.g. If M is a qubit and the measurement is boolean (2-valued) then the hidden measurement E collapses the Bloch ball to an axis.

We've described **perfect measurements**, given by homomorphisms from A to M. If f is not injective, then we can replace A by $B = f(A)$. In this context, first Bob measures M and then Alice measures Bob. The only way this is possible is if A has unused values. The actual measurement is from 1 to 6. But Alice register holds values from 1 to 10. So you can simplify the measurement so that the register only holds values from 1 to 6.

If $M = M(d)$ is a qudit then a classical $A = a\mathbb{C}$ only embeds in M when $a \leq d$.

Another description: If a qudit $M(d)$ or a larger $L(H)$ has a Hilbert space H and A is the finite register of a perfect measurement then the measurement comes from an orthogonal decomposition of H into subspaces H_k (k in A).

Say, $7 = 3 + 3 + 2$ (ternary measurement). 7-dimensional Hilbert space. Choose any 3-dimensional subspace. Projection onto that is the first boolean. In the 5-dimensional complement choose another 3-dimensional subspace. The remaining complement is 2-dimensional.

(Note that each boolean element of the algebra have a rank which corresponds to the dimension of the image space.)

Here b_k = perpendicular projection onto H_k. $|\psi\rangle \to b_k|\psi\rangle$ is also the unnormalized Bayesian posterior of b_k.

Imperfect measurements aren't given by homomorhisms at all. In the qudit case they are called POVMs. POVMs are measurements that allow noise blur.

We note that A can be *any set*. It doesn't have to a subset of $\mathbb{R}$. Real-valued measurements are used to help answer what is the system; not what measurements can we do. Stereotypical physicist says that they can't tell you whether it's an apple, banana or pear unless your number them! That doesn't make sense in the context of measurements.

If x is a real spectrum discrete operator then we can let A be the spectrum and then get a number-valued measurement where H_k is an eigenspace.

A TPCP from $A^\#$ to $B^\#$ might also be unital if we use trace to identity A with $A^\#$, say in the qudit case. If A and B are qudits and E is a TPCP $A^\# \to B^\#$ then the unital condition $E(1) = 1$ is equivalent to E preserving the uniform state, the center of A^Δ.

Classically E (regardless) is a stochastic matrix and then this extra makes E doubly stochastic. Quantumly, such an E is still called doubly stochastic.

What does a doubly stochastic map do? It doesn't make any negative probability, preserves total probability and most importantly preserves the uniform distribution. To preserve uniform distribution the row sums should also be 1 (in addition to column sums being equal). E.g. E can be a permutation matrix which is exactly doubly stochastic and deterministic.

1.4 Birkhoff's Theorem

Theorem (Birkhoff): Every doubly stochastic matrix is a convex sum of permutation matrices.

Example: $A = B =$ a bit, $E =$ bit flip with probability p

$$E = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}$$

If $A = B = M(d) =$ a qudit then unitary operators are doubly stochastic.

(1) Any convex sum of them is doubly stochastic and many TPCPs aren't. Visualize this in qubit case. An unitary or algebra automorphism will simply rotate the Bloch ball. The middle will still go to the middle. But then you can also throw away the qudit and recreate a state at the pole. It's perfectly realistic but not a convex sum of unitaries.

(2) The **quantum Birkhoff theorem** is true for qubits, any UTPCP (doubly stochastic) is a convex sum of unitaries but it's not true for qudits with d at least 3. The real flavour of Birkhoff's theorem is that it's a complete description of the extremals.

The question of extremal points of TPCPs is a notorious problem. Unitaries are simple examples of extremal UTPCPs. The extremal CP rays are easy – they are individual Kraus terms.

(3) Hidden measurements are convex sums of unitaries.

Decoherence due to a hidden measurement can be accounted for by multiplying each subspace with a scrambled phase factor.

1.5 Bell's Theorem

Theorem (Bell): QProb does not embed in Prob as a tensor category, not even approximately. Whereas Prob is immediately a subcategory of QProb. Anything like an embedding from QProb to Prob would respect the Bell inequality for two qubits.

Any category can be realized as a category of sets and functions between them. But that theorem is false if you state tensor categories. To put QProb in Prob you would have to repeal the concept of joint system and have a grand theory of codependence. Then you might be able to wish away quantum probability. Quantum entanglement even for two qubits cannot be modelled by classical systems, even infinite ones. You would have to able to split up a system into Alice and Bob to even have a theorem like this.

If you have just one quantum computer that you don't split into pieces you *can* model that using classical system. Though if you do the corresponding classical might be exponentially larger. By the way, we should note that closed system rules don't apply to measurement.

Note: The semi-quantum trit has more than one possible center, depending on what you're trying to do. The centroid may not be the maximum entropy point.

1.6 Models of Computation

Any tensor category supports circuits = Tensor networks with a time arrow.

Three fundamental cases for us:

1. **Deterministic circuits**: The category is Set. The tensor operation is Cartesian product. (AND, OR, and NOT are present here.)
2. **Randomized circuits**: The category is Prob. The tensor operation is tensor product of commutative algebras.
3. **Quantum circuits**: The category is QProb. The tensor operation is tensor product of the finite dimensional VNAs.

In (1), certainly bits $\mathbb{Z}/2\mathbb{Z}$ or 2-element sets in general are objects. AND, OR, NOT, and COPY (1 bit goes in and 2 bits go out) are morphisms. The first theorem as a tensor category, is that these gates generate a big part of Set (not all objects but many objects, and all morphisms). In fact, they generate all objects with 2^n elements and all morphisms in between.

Extra trick, the Karoubi envelope trick for any category. Any category C has a Karubi envelope, where for each object A and each idempotent $f^2 = f$ on A, (A, f) becomes a new category.

If we throw in the Karoubi envelope as part of "generate", the standard gates (*) generate all of Set.

The purpose of this is to make sets of each finite size from just sets whose size is a power of 2. At first, I can only make sets whose sizes are powers of 2. How do I make a set with 3 elements? Well, choose a mapping from 4-elements (2 bits) to 4-elements (2 bits) such that that the stray 4th value to send it to one of the other three. So you empower this 4-element set with a correction demon and the correction demon just sends the 4 bitstring values back to 0.

$$\text{Karoubi Demon}\colon 00, 01, 10, 11 \to 00, 01, 10, 00$$

Register coerces the values to trit values. Demon sends $3(= 11)$ back to $0(= 00)$.

It's a way to make new objects from old objects in any category. This is a object factory.

Same principle as for finitely generated group. That the finite generating set of the category set doesn't matter much. You can interconvert with a finite overhead.

P/poly = non-uniform polynomial time is a set of functions (or sequences of them) that have poly-sized circuits.

For randomized computation, there's a problem that's going to continue in the quantum case. The set of morphisms $\mathrm{Hom}(A, B)$ is uncountable but a finite set of gates can at best reach, at best, a countable number of them. The morphisms are stochastic maps and have continuous parameters.

Two solutions:

(1) (Less popular) Allow a continuous family of gates.

(2) (More popular) Allow circuits to approximate rather than equal a target stochastic map, i.e., densely generate.

Produce for me a coin flip such that the probability of heads is $\frac{1}{e}$. With fair coin flips you can only create diadic rationals where denominator is some power of 2.

Theorem (AND, OR, NOT, COPY and "RANDOM" = random bit creation, 0-ary gate with 0-input and 1-ouput), together densely generate Prob. We still Karoubi envelope trick as we to generate objects whose dimensions are not powers of 2.

This lets you slide back from Bayesianism to frequentism. The random bits can be viewed as certificate that are helping the computation (frequentist interpretation).

2 Lecture 2 (19 November 2021)

2.1 Measures of Fidelity

$E\colon \mathcal{A}^{\#} \to \mathcal{B}^{\#}$ is a desired quantum map. You might instead see $F\colon \mathcal{A}^{\#} \to \mathcal{B}^{\#}$.

1st for states $\rho, \sigma \in \mathcal{M}^{\triangle}$:

$$d(\rho,\sigma)\colon = \max_{b \in \mathcal{M}_{\mathbb{Z}/2\mathbb{Z}}} [\rho(b) - \sigma(b)] \overset{\text{Thm}}{=} \frac{1}{2}||\rho - \sigma||_1$$

This is called trace distance, infidelity (sort of) or variation distance.

$\frac{1}{2}||\rho - \sigma||_1 \overset{\text{Thm}}{\Longrightarrow}$ Same bands for general distinguishability for ρ vs. σ or E vs. F for any use with only one copy.

$d(E,F)\colon = \sup_{\rho \in \mathcal{A}^{\triangle}} d(E(\rho), F(\rho)) \overset{\text{Thm}}{\Longrightarrow}$ Same bands for general distinguishability for ρ vs. σ or E vs. F for any use with only one copy.

Worst case infidelity,

Theorem $d(E \otimes G, F \otimes G) = d(E,F)$. Contrast TPP vs. TPCP. Also contrast ensemble fidelity vs. entanglement.

$$d(E_1 \otimes E_2) \leq d(E_1, F_1) + d(E_2, F_2)$$

$$d(E_2 \circ E_1, F_2 \circ F_1) \leq d(E_1, F_1) + d(E_2, F_2)$$

2.2 Karp-Lipton Theorem

Karp-Lipton Theorem: $\mathsf{P/poly} = \mathsf{P}_{\text{non-uniform}}$

$\mathsf{P/poly}$ represents a Turing machine with a polynomial time budget and and polynomial advice from an angel. $\mathsf{P}_{\text{non-uniform}}$ represents sequences of poly-sized circuits.

The $\supseteq$ containment is thought of as angel providing the circuit whereas the $\subseteq$ containment is an unrolling argument.

Theorem: $\mathsf{P} = \mathsf{P}_{\text{uniform}}$

P represents polynomial sized Turing machines whereas $\mathsf{P}_{\text{uniform}}$ represents circuits drawn by one polynomial-time algorithm.

The $\supseteq$ containment is thought of as simulating your own circuit. The $\subseteq$ containment is an unrolling argument.

2.3 Tensor Circuits

Tensor networks in suitable $\otimes$ category gives you circuit computation:

Objects	Maps	$\otimes$	poly-sized circuits
Set	functions	$\times$	P/poly
Prob	stochastic	$\otimes$	BPP/poly
QProb	TPCP	$\otimes$	BQP/poly

Fact: In all 3 cases, you get correct P, BPP or BQP in one of two ways:

1. TM draws a circuit. 2. Use periodic circuits (special case of 2) = cellular automata (Fig. 1)

Each category has generating sets, except, for Prob and QProb you need dense generation. You need a Karoubi construction (make new objects with a Karoubi coercion idempotent map) to get all objects instead of just $(\mathbb{Z}/2\mathbb{Z})^n$.

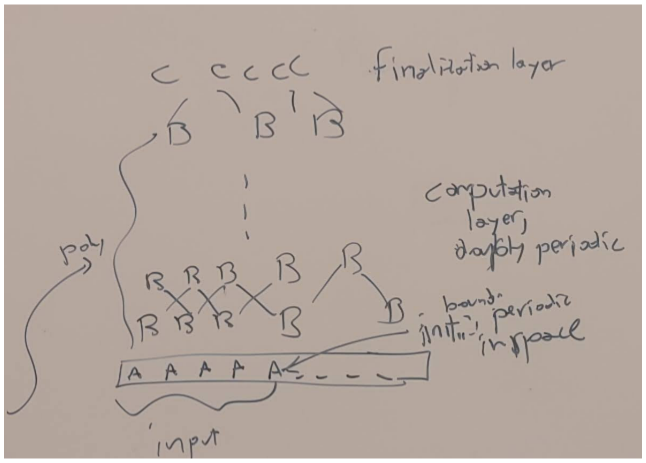


Figure 1: Periodic Circuit

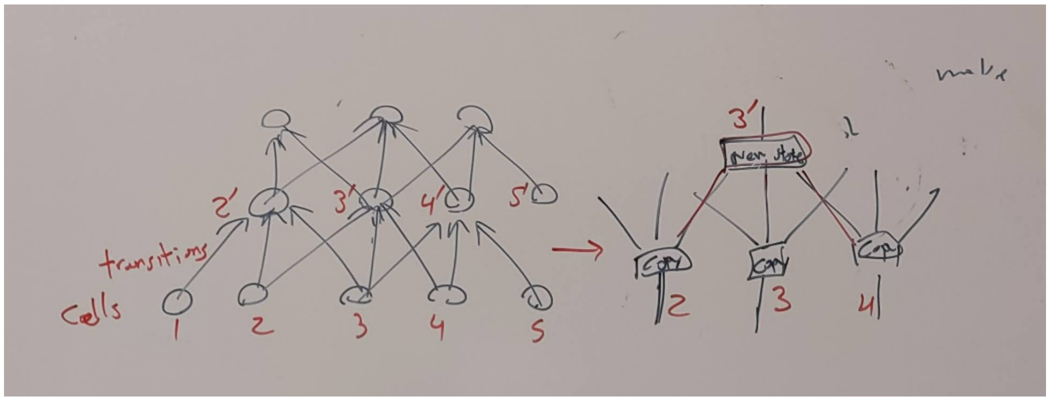


Figure 2: Computation by an Automaton

P/poly: $\mathbb{Z}/2\mathbb{Z}$, AND, NOT, OR, COPY is used to generate all objects but it doesn't matter.

BPP/poly: Random source factorization. We determine gates to any 0-ary random bit gate = generator. (**Theorem**: This kind of generation is dense.)

BQP/poly: Stinespring dilation. Promote all bits to qubits (except at the end!). Promote all TPCPs to unitaries + initialize fresh ancilla qubits in $|0\rangle$ states.

We said dense generation. In both cases, there is the *efficient* dense generation problem. To express my gates in your gates you need larger and larger approximate circuits and that should be uniform.

Necessary condition: The parameters in the gates should be efficiently computable numbers. Then there's a **theorem** saying that efficient generation is possible. And, there exists infidelity with a polylog(ϵ) overhead. This is basically the statement of the **Solovay-Kitaev theorem** in the quantum case.

2.4 Computation by Automaton

Figure 2 illustrate of how computation is done by an automaton.

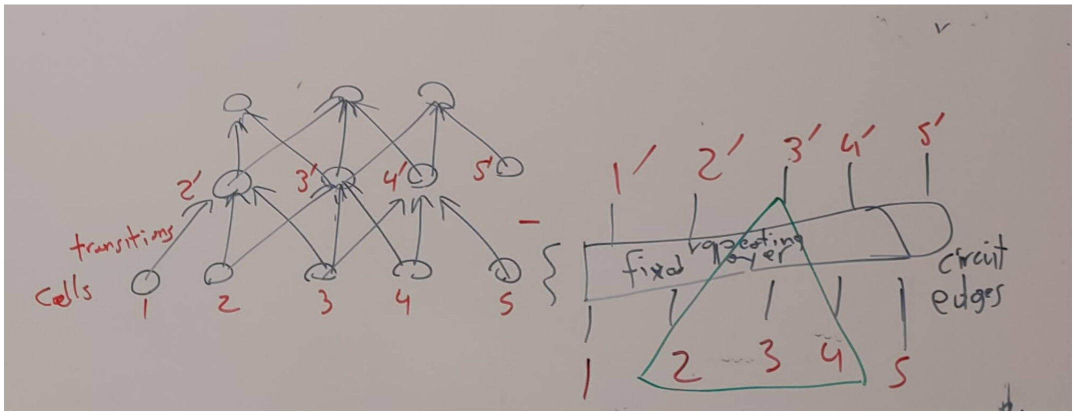


Figure 3: Representing Periodic Circuit as an Automaton

3 Lecture 3 (30 November 2021)

3.1 BQP (Bounded-error Quantum Polynomial)

1. Functions **computable by polynomial sized circuits** (uniform families or even just periodic circuits) of TPCPs.
2. **Circuit cleanup** - Using gate-by-gate Stinespring dilations can reduce to
a) Unitary gates
b) Ancilla initialization
c) Measurement at the end
3. Reasonably intuitive, realistic extension of (2) is a **classical TM with a quantum tape**.

The B, by analogy with BPP means bounded error probabilistic. Shoehorns Prob/QProb models into framework of deterministic questions. Input is classical deterministic.

$$\Pr(\text{Correct answer at the end}) > \frac{2}{3} \text{ (say)}$$

It's even better to say $> 1 - \epsilon$. $\operatorname{poly}(|\text{input}|, \log(\epsilon))$.

3.2 QFT vs. DFT

QFT vs. DFT on $\mathbb{Z}/2^n\mathbb{Z}$ Input: n qubits vs. 2^n floats. Performance: $\operatorname{poly}(n)$ vs. $\mathcal{O}(N \log N)$ where $N = 2^n$.

$$\mathbb{Z}/2^n\mathbb{Z} \hookleftarrow \mathbb{Z}/2^{n-1}\mathbb{Z} \hookleftarrow \mathbb{Z}/2^{n-2}\mathbb{Z} \hookleftarrow \mathbb{Z}/2^{n-2}\mathbb{Z} \hookleftarrow \dots$$

3.3 Basic Complexity Classes

Figure 4 shows a chart of some fundamental classical and quantum computational complexity classes.

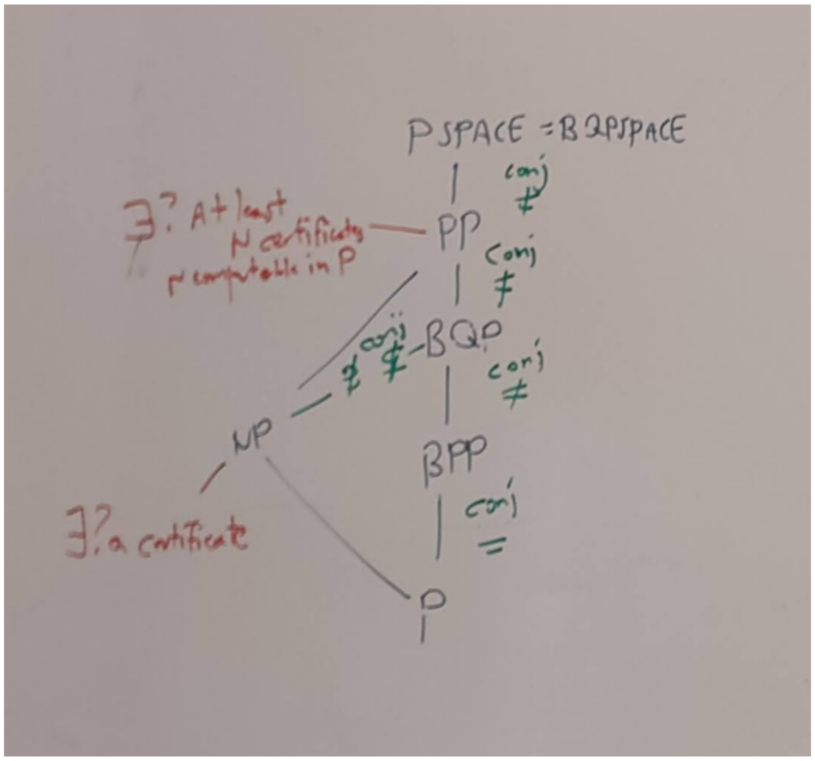


Figure 4: Some Fundamental Computational Complexity Classes

4 Lecture 4 (7 January 2021)

4.1 Review of measurement in quantum mechanics

Suppose that (b_k) is a partition of unity by booleans in the algebra $\mathcal{M}$ indexed by the set A, i.e., mutually disjoint booleans that add to 1. For any boolean b, there is a corresponding Kraus term B acting on $\mathcal{M}^\#$ (note the sharp notation for the predual rather than the dual).

For any VNA $\mathcal{M}$, you can break $\mathcal{M}$ into 2×2 block matrices as follows:

$$\mathcal{M} \cong \begin{bmatrix} b\mathcal{M}b & (1-b)\mathcal{M}b \\ b\mathcal{M}(1-b) & (1-b)\mathcal{M}(1-b) \end{bmatrix}$$

Here $b\mathcal{M}b$ and $(1-b)\mathcal{M}(1-b)$ are the corner terms, and $b \in \mathcal{M}_{\mathbb{Z}/2\mathbb{Z}}$. The off-diagonal elements are not Hermitian states but other superoperators (pre-dual elements). They are not normalized but normalizable or they vanish.

$$B(\rho)(x) \coloneqq \rho(bxb)$$

Let's recall the definition. $\rho(x) \coloneqq \mathrm{Tr}(\rho x)$. Works for all type-1 VNAs. You can express all predual elements as matrix themselves using this formula.

Then $B(\rho) \coloneqq b\rho b$. This is directly a Krauss term. Given the formula $\rho(x) \coloneqq \mathrm{Tr}(\rho x)$ you can interchange between $B(\rho) \coloneqq b\rho b$ and $B(\rho)(x) \coloneqq \rho(bxb)$. To normalize or rescale probability, $\rho'(x) \coloneqq \rho(bxb)/\rho(b)$. But this is non-linear in ρ. The linear filter on states is the one where you scale the probability back up when you have confirmed that the boolean is true.

If you imagine a probability distribution on people or die rolls. Then you have a Boolean event. We choose a person from the class. Is the person male? Then the total probability is only the chance that the person is male. In other words, the posterior probability distribution is just the restriction to the male in the class. The total probability is no longer a 100%. That has the advantage of being a linear map on probability distributions.

So the operator algebraists would say CP maps. Since the physicists are specifically thinking of matrix algebras they would say superoperators.

Anyway, $B(\rho) \coloneqq b\rho b$ is simply the unnormalized posterior state formula, if the boolean b is true. Now we're just using that formula as a CP map or a Krauss term itself and assembling another CP map, which is TPCP, by adding up this thing we selected for all of the booleans in a partition of unity.

The unnormalized posterior is the only thing that's consistent with classical posterior state. There's an argument to that. If you have finite matrices, one way to make everything classical is to make all states and observables diagonal, then what you see this formula is doing: it's just chopping to the domain where the boolean b is true.

If you open a physics book, what do they mean by boolean? Well, they want all their random variables to be real valued. Then what is the physicists formula for the state once you have measured a given eigenvalue of your operator: if b is the eigenvalue projection then $|\psi\rangle \to b|\psi\rangle$ unnormalized. This, however, is not our definition as a fan of mixed states and operator algebras. This definition is something that Heisenberg and Max Born could have written down just as well and they did!

What does ρ look like when it's predual to a matrix algebra and it's pure? Then it has the form $\rho = |\psi\rangle\langle\psi|$. Then $b\rho b$ is $b|\psi\rangle\langle\psi|b$. We have just demonstrated this is entirely consistent with our formula $B(\rho) \coloneqq b\rho b$. It's the same thing written in bra-ket form instead of ket-bra form. To translate it, think of it as $\langle B(\psi)|x|B(\psi)\rangle = \langle\psi|bxb|\psi\rangle$, and by golly we're just doing the b-eigenspace projection s.t. $|\psi\rangle \to b|\psi\rangle$ (b = a boolean, or a self-adjoint idempotent).

Now when you have a partition of unity: $1 = b_1 + \ldots + b_n$ and $b_i b_j = 0$ for $j! = k$. What's being said is that every alternative is covered and you *will* measure one of these answers and exactly one.

If you express 1 ... in mathematics partition means you're a set, number or a function. Partitioning a function is like partitioning a number. Say, $13 = 7 + 3 + 3$. (By the way, a sum of booleans adding to 1 must be disjoint.)

A single big B is CP, as in $B(\rho)$. There's a formula to make a TPCP as well. If $1 = b_1 + \ldots + b_n$ but then $E_h = B_1 + \ldots + B_n$ is TPCP, not just CP. Because of completeness total probability is conserved/preserved. This is a kind of Krauss term decomposition, given that each large B is a Krauss term. (E is demanded to be a TPCP a priori and we're fulfilling that demand by giving the formula for it.)

The problem for this E is that the measurement outcome is not written anywhere. So it's actually the TPCP for hidden measurement. What is does in general to states is that it will change the states to a block diagonal restriction (it will zero out the off-diagonal blocks). So you will keep $b\mathcal{M}b$ and $(1-b)\mathcal{M}(1-b)$. It's Krauss' theorem for matrix algebras that the extremals of the CP cone are precisely Krauss terms (not necessarily booleans) - terms that are linear on pure states. This E basically produces a posterior state - the measure is a spy who didn't tell you what they observed. Read more on this Wikipedia page: Binary measurements.

There are other TPCPs for destructive and non-destructive measurement.

$E_v = B_1 \otimes [1] + B_2 \otimes [2] + \ldots + B_n \otimes [n]$ models "non-destructive" visible measurement – measured value and posterior state. $E_d(\rho) = \rho(b_1)[1] + \ldots + \rho(b_n)[n]$ is destructive measurement (read more here). Interpret $\rho(b_k)$ as the chance or probability of outcome b_k; it's the chance that b_k is true. If we throw away the posterior state in E_v we get E_d and if we throw away the measurement value from E_v we get E_h. [Note: ρ(boolean) is the probability that the boolean is true and we built everything around that.]

If A, the classical measurement register is discrete, then all three of E_h, E_v and E_d exist. Otherwise, in general if A is anything classical, only E_d exists. There's a way to define E_d that does not require A to be discrete: A can be anything commutative and you can take a von Neumann algebra homomorphism. It just means that you've measured things too much you've kind of pulverized things too much to have a posterior state. As an example of this phenomenon, say we're measuring the exact position of a quantum particle, where the wavefunction is an L^2 function on the reals. If you have an L^2 function on the reals, you can convert it to a probability distribution on the reals by taking the square norm and not integration. But there can't be a posterior state because it would be a delta function at a single measured value – and those aren't normalizable states. Something similar goes wrong in the VNA formalism but not completely (when A isn't discrete). The destructive measurement still exists (kind of notionally). These idealized real measurements aren't real life. Real life always has noise limits and information finiteness of some kind or another.

What the physicists do is that they will have a Hilbert space for a quantum system because they do want to talk about some von Neumann algebra of observables acting on that Hilbert space. Maybe all the bounded operators or maybe just some of them. So they will step outside of that Hilbert space and discuss other states. Then having stepped outside they'll have the warning that this one is not normalizable. What they'll mean by that is that the actual realistic states will only be approximations to these idealized things. If you have a particle and you've fully measured its position – it's posterior state is a delta function. They'll say it means it's just a very localized state. They just won't be rigorous about that. But they actually referring to concepts like approximate eigenvectors of continuous spectrum operators.

(If the terms b are minimal booleans then for visible measurements you're collapsing to one *particular* state.)

$z|\psi\rangle$ for $z \in \mathbb{C}$ is a line of states. Unnormalized but normalizable. When you put such a thing in density matrix form it becomes a real ray because $z\overline{z} = |z|^2$.

4.2 Brief review of VNAs and C* Algebras

When we say that von Neumann algebras are C* algebras with preduals it's Sakai's definition, not von Neumman's. von Neumann defined his algebra from a Hilbert space representation. Then he says: "Here's why the choice of Hilbert space doesn't matter. It's just scaffolding". Sakai discovered a way to skip the scaffolding entirely. So we prefer that for philosophical reasons.

von Neumann wanted to say that if we have a represented C* algebra then it's a VNA when it's closed under the weak operator topology. For any Banach space, any predual, which may not exist, or if it exists may not be unique. But any predual is (theorem) present as a closed subspace in the dual and in fact you can define that way. That is, not just any closed subspace in the dual but a favourable closed subspace in the dual. There is an abstract and concrete definition of a predual of a Banach space.

Abstract definition of a predual of a Banach space X: Some Banach space Y that you found somewhere together with a chosen isomorphism $Y^* \cong X$.

A more concerete definition: A predual Y of X is a subspace Y of X^* such that the evaluation map from X to Y^* is an isometry. In other words, interpreted in reverse form, from X to Y^* is an isometry.

If Y is any subspace of X^* then Y of X is defined: Y is a linear map from X to $\mathbb{C}$. That is the definition of dual vector. So we reinterpret Y of X and X of Y. It's like flipping bras and kets really. This induces a linear map from X to Y^*, which in favourable cases is an isometry between Banach spaces.

In the abstract definition, Y is floating somewhere and its dual is matched to X. In the concrete definition, Y lives in a specific home in X^* but it is a favourable closed subspace of X^*. It has to be Banach space itself so it has to be closed. The sonly ubspaces of Banach spaces that are themselves Banach spaces with the same norm are closed subspaces. Then after that, well, the fact that you did pick a subspace of X^* means that you get a linear map from X to Y^*, using this clever definition that switches the role of bras and kets. And if that happens to be an isometry, then congratulations, you found a predual inside the dual.

Theorem: The two definitions are equivalent.

In the concrete definition, the subspace was mandated to closed, because Y is supposed to be Banach. And not with some completely different norm. If we were allowed to change the norm, then Y could be Banach with some other choice of norm. Keeping the norm the same, for Y to be Banach, it has to be closed. The part that's a theorem and not a fiat, is getting from the abstract definition to a concrete defintion.

Now if M is von Neumann then it has both a dual space which is its all of its states (if you interpret M as a C*-algebra) and just the predual states which are nicest ones. There is a piece of wisdom, that the ... operator algebraists overload the word normal – they call the predual states normal. The piece of wisdom is that the non-predual states are wild. They fail to be continuous with the coarsest important topology on M. And their weaker continuity properties means that you can't do probability theory with them in the same way. And also you may need the axiom of choice to find any states to find any states that aren't predual. We think if you have infinite dimensional

Hilbert space, and let M all the bounded operators, the predual states are states that you can find – trace class operators and you can find matrices of them. The ones that aren't predual? We think that we need to rely on the axioms of set theory to know that any non-predual states exist for all bounded operators in an infinite dimensional Hilbert space. You will never find a formula for such a state. They do have some interesting properties but you're wandering into a regime of analysis that intersects with set theory issues. These states just don't have the same relevance to physics. Anyway, in finite dimensions, predual = dual, so we don't need to split hairs there.

5 Lecture 5 (14 January 2022)

5.1 Brief introduction to topological quantum computing

There is a dual role of categories and category theory, including in QCQI:

1. Use one category to describe a research area. E.g., group theorists do research in the category of groups. Quantum information theorists do research in the category of VNAs with TPCPs (quantum maps) as morphisms.

2. Use several or many categories within an area of research as objects of study.

Example: The category model for anyonic statistics. A closed system of anyons forms a category, where crucially the fusion of two anyons makes another anyon in the same category, although which may involve a quantum measurement.

A fusion category (for anyons) is an example of a tensor category, which to review is a category with a multiplication law "$\otimes$" on objects and morphisms. Also want a "$\oplus$" law to model mutually exclusive possibilities for the system. In QCQI, $A \otimes B$ means "Alice and Bob", $A \oplus B$ means "Alice or Bob". (I found a very clear explanation of the difference between $\otimes$ and $\oplus$ in the context of quantum mechanics, here: physics.stackexchange.com/a/528445).

For anyons, the Hom spaces are Hilbert spaces, that are used to describe topological state, the objects are just abstract and can't be vector spaces usually.

Example: Fibonacci anyons, in the Fibonacci category.

In the Fibonacci category, there are two irreducible particle identities: I = boson and F = Fibonacci anyon. I = boson and F = Fibonacci anyon.

$F \otimes F \cong F \oplus I$

If you solve for $\dim F$ from this formula, you get the golden ratio: $(\dim F)^2 = \dim F + 1$.

5.2 GNS construction: an important baby case

Say $M(d)$ is a qudit and ρ is a state. Does it have a purification $|\psi\rangle$? I.e. it is the marginal state of $|\psi\rangle\langle\psi|$ on $M(d) \otimes M(a)$ for some a? Maybe $M(d) \otimes M(d)$?

Direct explicit approach:

Make $M(d) \otimes M(d)$, solve for $|\psi\rangle$ from ρ.

GNS approach:

Take just $M(d)$ and ρ and make a new Hilbert space from them which has dimension ad. Hilbert space is built around $|\psi\rangle$, actually.

Since ρ is positive, $\rho(y^*x)$ is a positive semi-definite inner product on $M(d)$ as a vector space. So there is a ρ-seminorm $||.||_\rho$ on $M(d)$ coming from a Hermitian inner product.

$\langle y|x\rangle : = \rho(y^\dagger x)$

If ρ has full support, then $M(d)$ (in finite dimensional case) is just then a d^2-dimensional Hilbert space. If ρ does not have full support, you divide $M(d)$ by the kernel of $\langle y|x\rangle$, to get a da-dimensional Hilbert space if $a = \mathrm{rank}\rho$. Then $|1\rangle$ becomes a purification of ρ.

$|1\rangle \in \mathcal{H}_{M,\rho}$ where $\mathcal{H}_{M,\rho}$ is a Hilbert space made from $M = M(d)$ corresponding to $1 \in M$.

If $\dim M$ is finite, then $\mathcal{H}_{M,\rho} = M$ as a vector (or is a quotient if ρ has limited support).

If $\dim M$ is infinite, then M is only a pre-Hilbert space and you should take its completion to get H.

M acts on $\mathcal{H}$ in such a way that you can later check $\langle 1|x|1\rangle = \rho(x)$ for x in M.

$\mathcal{H}$ is M reinterpreted, maybe completed, action of M on H is just action of M on M by left multiplication.

ρ interpreted as creation of ρ ex nihilo as a TPCP from $\mathbb{C}^\Delta$ to M^Δ. Purification is a special case of Stinespring, and actually Stinespring's theorem can be proven GNS style.

Note: nLab has a similar explanation which might help to clarify some things.

6 Lecture 6 (21 January 2022)

6.1 Practical aspects of Stinespring factorization

$\mathsf{P}, \mathsf{BPP}, \mathsf{BQP}$ (and their counterparts for other amounts of time and space) come from three tensor categories. We can make circuits in any tensor category, with a few extra properties we have a definition of computing or approximating a decision function on bit strings. Or a function from bit strings to bit strings.

There is the problem that the 1st draft of a function on any-length bit strings can be a "non-uniform" sequence of circuits. As length of input grows you need a different circuit. In the naive definition, you can do something uncomputable with the choice of circuits as the length of the input changes. So there are two roughly equivalent solutions (equivalent for polynomial time):

1. The circuits are drawn in deterministic P in polynomial time.

2. (Quantum case) A supervised adaptive circuit drawn by a probabilitic TM which is then equivalent to a probabilistic TM with a qubit tape.

(1) is a special case of (2). Because the Turing machine that manages the qubit tape can just execute the gates blindly on the tape, until at the very end when it's time to measure the qubits to get the answer.

3. Periodic circuits.

(2) generalized (1) which generalizes (3).

But circuits of what? In all cases, the category has a generating set of objects and generating morphisms = gates.

Use the "Karoubi envelope" trick to make a full set of objects. If you have an operation that's an idempotent on bitstrings you can use that as a model for digits whose size isn't a power of 2. Any time you have idempotent operation that's in category, for any computation model, you can view that operation/object as a new type of object together with coercion of values into new type of object you wanted. For instance, the same mechanism that let's you make a ternary digit out of two bits lets you make a classical bit out of a qubit, just with a different choice of projector.

Once we decide what all the morphisms are. Though the computer science instinct is to first declare the generators and see what they generate. In pure mathematics, we first declare the full range of what's allowed and then worry about generators later. A lot of questions don't depend on the specifics of the generating set anyway.

What are the morphisms?

For deterministic computation: function between finite sets. For classical probabilistic computation: stochastic maps. For quantum computation: TPCPs.

Why are TPCPs the correct morphisms?

1. If a map from A^Δ to B^Δ is not TPCP then for one reason or another it can't be realistic. Either it violates additivity of classical probability or it doesn't conserve probability or it creates negative or non-real probabilities. Maybe after tensoring with an entangled bystander doing nothing.

2. If the map E is TPCP then it factors into a composition of stages that are accepted as realistic. Any such factorization is a Stinespring type theorem. (Nicest way to prove it is by a GNS-type construction.)

The three factors of a the relevant Stinespring-type factorization:

(2a) Create a bystander Hilbert space $\mathcal{H}_C$ in standard vector state $|0\rangle$. This must be assumed to be realistic. If you didn't believe this were realistic you wouldn't believe quantum mechanics at all. (C the algebra is $L(\mathcal{H}_C)$.)

(2b) Apply (the predual of) an automorphism of $L(\mathcal{H}_C) \otimes A$. The precept here is that you're allowed to rename your instruments. You haven't even changed the system in some sense. (How do you rotate a house without moving anything? Just rename your North and South!)

The renaming does not preserve locality though and it may destroy the tensor structure. The system was split into two and the renaming may not respect the way it was split into two. In physics and computer science, let's say you have 3 bits. You are allowed to rename them as the digits from 0 to 7. But if you do that renaming you're not respecting bit locality. This is controversial in physics.

(2c) Pick a specific renaming where you can identity a subalgebra of the observables as being isomorphic to the observables in B. You can find an isomorphic copy of B as a subalgebra of $L(\mathcal{H}_C) \otimes A$, such that the composition of these three stages is your chosen TPCP E. All E had to be to make this factorization possible is TPCP. Then you say that we're not interested in the other instruments and we'll just keep these. And that's absolutely realistic. That's just saying that if you have a toolkit of instruments you don't have to use all of them.

The purpose of the renaming step (2b) was to find a subalgebra of $L(\mathcal{H}_C) \otimes A$ that is literally B rather than just isomorphic to.

The basic argument is that if E is TPCP then it's realistic. Though the locality issue has to be taken more seriously if we're doing quantum gravity. At the level of QCQI or condensed matter physics (where we can at least apply automorphisms that are unitary operators; that's a basic necessity) this argument is completely conclusive.

So the Stinespring factorization in the practical rather than philosphical viewpoint lets you first locally (for gates) and then globally (for the whole circuit) factor the circuit into:

1. Initialize ancilla qubits to 0.
2. Apply unitaries.
3. Measure the results, or toss qubits.

That's not a philosophical thing but also a practical matter. It's also important pedagogically. It is important to realize that algorithm technique can be put into this partially canonical form. This is part of quantum algorithm design often though not always. It shows that the TPCP definition is equivalent to the unitary gate definition. Because the factorization is algorithmic and the whole thing is a deterministic polynomial time calculation in principle.

It's kind of disappointing actually. Every bit is promoted to a qubit. Also an ancilla bit is usually not needed until the 11th hour but it created at the beginning. So anything polynomial time and polynomial space is wrecked by this kind of thing. Nevertheless, it is all polynomial time because you just do this for each gate separately and pull all the ancillas to the beginning and all the measurements to the end.

This practical use of Stinespring is equivalent to random source factorization of Stochastic maps. Any Stochastic map factors into creation of randomized ancilla coin flips followed by a deterministic map. This answer a small philosophical question that says all stochastic maps are realistic because if you believe flipping coins is realistic, you can make any stochastic map. And if you believe that deterministic circuits are realistic. But as a practical matter and as a pedagogical matter the classical randomized analogy is also important. It lets you say that instead of reinventing the entire model of computation, in stochastic terms, instead of thinking of randomized algorithms as a machine that processed probability distributions, you can outsource all of the randomness to a source of randomness that's separate from the deterministic algorithm.

This source of outsourcing and importing randomness was crucial for understanding and solving a crisis in internet security. There was no favourable source of randomness for cryptographic keys.

The first web browsers just called PRGs based on booting time but that was too little entropy. Then they used key stroke timing and then later the modern world shifted to noise generated by heat sources. Nowadays CPU chips have quantum or thermal orgin. You really need a source of randomness for stochastic computation. (We can view the coin flips as certificates in the same sense as NP. You do not have this dual interpretation in the quantum case. There is some decent analogy between Stinespring factorization and coin-flip factorization in any case.)

6.2 Introduction to Solovay-Kitaev theory and Kuperberg's modification

BPP (and its category prob) and BQP (and its category prob) have continuous parameters, but in the healthy version finite gate sets which only *densely* generate. Then there is the efficient dense generation challenge. My gates can approximate your gates but can I quickly compute (in P, preferably) a low-overhead approximation? If the gate set matters then my gate set might be adapted to your algorithm.

So in P the question doesn't exist. In BPP (or prob), the question reduces to making a fair coin flip from an unfair one, or vice versa, at least approximately. You can't get exactly $\frac{2}{3}$ but you can get very close to $\frac{2}{3}$ by expanding in binary. Is it efficient? Usually yes, just require logarithmic number $\log(\epsilon)$ number of steps for precision ϵ and an algorithm to get the digits of p itself.

There was a famous protocol suggest by von Neumann. If you want a fair coin flip from an unfair coin, you flip the coin twice. If the two answers agree then you throw them away. However, if one is heads and another is tails you use the value of the first coin as the answer.

In either prob or qprob, "poly-time" means really $\mathsf{poly}(|\text{input}|, \log\epsilon)$.

The quantum version of the $\log\epsilon$ coin flips is the Solovay-Kitaev theorem and the Solovay-Kitaev problem. If you can do what you wanted for a single coin flip then the solution spreads to any bounded number of coin flips. If you wanted a particular probability distribution on a bounded set then you can divide the process of making that probability distribution into several coin flips.

Let's say you wanted a particular probability distribution on the numbers from 1 to 5 and you just have 5 particular target probabilities. You can have one coin flip for whether the answer should be odd or even. Let's say it's even. Then you can a second coin flip for 2 and 4. If it's odd then a second coin flip to separate 1 from 3 and 5. And finally a third flip. As long as it's a bounded size probability distribution then you've reduced it to a bounded number of coin flips and that's just a constant extra factor of work and then you're back to the $\log\epsilon$ thing for individual coin flips.

The quantum version of the $\log\epsilon$ coin flips is the Solovay-Kitaev theorem and the SK problem.

Problem: Suppose you have a finite set of 1-qubit gates that densely generate $SU(2)$, or people really care about $PU(2)$. You can use the fact that global phase doesn't matter to set the determinant of the unitary to 1. But it's a little bit closer to interpretation to say that you should just take the whole unitary group and quotient by phase and get $PU(2)$. We know $PU(2) \cong PSU(2) \cong O(3)$. You have an algorithm to quickly calculate the matrix entries of your gates and you want an algorithm to generate an economical approximation to any target unitary A, again with assuming that you calculate digits of matrix entries of A. You want to approximate any A in $PSU(2)$ (say) with gates, with efficient calculation of matrix entries.

You should notice just how analogous the hypothesis of the theorems are – really based on the unitary factorization we already mentioned. We want to approximate any TPCP but if we're able to approximate the unitaries we're done, due to factorization. It's just a harder quantum version.

Theorem: $\text{poly}(\log\epsilon)$ computation time and gate number suffice for ϵ precision approximation to A. If you're allowed to use exponents written in binary to abbreviate the output word then polylog computation time wouldn't necessarily imply polylog gates since you could have a huge exponent.

The distinction is moot if you have to spell out all the gates explicitly; then polylog computation time implies polylog gates. The distinction in the other direction is never moot. Even if it's small number of gates, it could be as hard as NP complete to find the correct gate sequence.

In his solution to this question, Solovay compromised. Just counting suggests that the number of gates probably only has to be linear in $\log(\epsilon)$. But nobody has a solution, for the general case, that that's good. In order to find the sequence findable, it's longer than just linear.

There's two things going on. What the quantum computer does and what the classical controller does. The classical controller's resources are far cheaper. You can view it in real-time dynamical terms such as deciding what the next gate the qubits should go through. So the first polylog is classical controller's computation time to decide what the quantum chip should do. The second one is a much more important resource estimator, i.e., the number of quantum gates that will be executed. In one published solution, the second cost-bound (the more important one) by Dawson-Nielsen is $O(|\log \epsilon|^{3.9\cdots})$.

In Kitaev-Shen-Vyalvi, it's $\tilde{O}(|\log \epsilon|^3)$. $\tilde{O}$ means together with log factors in the denominator. They write it as $O(|\log \epsilon|^{3+\delta})$ for any $\delta > 0$. Written as $O(|\log \epsilon|^2)$ in Nielsen and Chuang which is wrong.

Theorem (Kuperberg): Just by simplifying KSV you can get exactly $O(|\log \epsilon|^3)$ for word length.

All of these algorithms use multi-scale approximation. Move A to an element near the identity and after that move it closer and closer in a multi-scale iteration with words in the gates that are close to I. You view the matter backwards. It's easier to discuss parking a car than unparking a car. Reaching a gate A is like parking a car though it will use some logic about unparking a car.

All of the above, uses the lemma that if X and Y in $SU(2)$ are $X = I + O(\epsilon)$ and $Y = I + O(\delta)$ then $[X, Y] = XYX^{-1}Y^{-1} = I + O(\epsilon\delta)$. This is a group commutator, not an algebra commutator which isn't a circuit sequence.

This is used to make multi-scaled correctors at scales $r, r^2, r^3, \ldots$ for some $0 < r < 1$.

Here Solovay was clever. What does not work: Make the order n corrector from a order $n-1$ corrector and an order 1 corrector. You can quickly see that this gives you exponential length words. The later corrections are finer and finer correctors.

What does work: Make the order n corrector from an order $\lceil n/2 \rceil$ and an order $\lfloor n/2 \lfloor$ corrector. If you do this the previous way, the previous corrector is almost is as expensive and you need two of them. Here the correctors are halfway back and you need four of them. So you reach twice as many stages with four times the cost. Then (roughly speaking) $O(\epsilon\delta)$ is just an O, but what we need is a Θ (better, as it builds correctors at different scales and different directions). This whole conundrum is the reason that the exponents were ever more than 3.

Then, roughly speaking, the corrector at scale r^n has word length $O(n^2)$. We also need to worry about how many times the correctors would be needed at each scale. Anyhow, let's just sweep that under roughly speaking for now. And then summing over all scales you get $O(n^3)$ and thus that $O(|\log \epsilon|^3)$.

Versus the lower bound: $\Omega(|\log \epsilon|)$ for word length, from just counting or estimating volume or entropy.

Amazing result of others: For special gate sets, including some of the standard ones, you can get a word length of $\Theta(|\log \epsilon|)$. Totally different algorithm and not as general.

Kuperberg's Theorem: You can break the cubic barrier.

Because, there are more efficient re-scaling words than $[X, Y]$. For instance, $[[X, Y], [Y, Z]]$ has length 14 and not 16. Instead of doubling precision at 4 times the cost, you can qudruple precision at 14, not 16 times the cost. This also has a two letter form $\text{length}([[X, Y], [Y, x]]) = 14$. The authors boosted these ideas as well as they could from 2 to $\log_4(14)$ to $\log_{\text{GoldenRatio}}(2)$ (Elkasapy and Thom). The total exponent is then $\approx \log_{\text{GoldenRatio}}(2) + 1$.

But (1) they are conjectured that this is optimal, (2) these are abstract self-correcting words that work in any Lie group. Maybe in $SU(2)$ specifically you can do better?

There are 274 trillion words of length 30.

Ideas:

1) Algebraic abstraction: Write $X = I + \Delta X, Y = I + \Delta Y$, and compute with truncated power series, where all of the coefficients. The inverses look like:

$$x = I - \Delta X + (\Delta X)^2 - \dots.$$

2) Store the expansion for half-length words and find pairs that are equal. This second idea is the main source of acceleration.

3) Homomorphism and p-adic Schwartz-Zippel. Map ΔX and ΔY to random integer matrices of some size, say 2 in the case of $SU(2)$, which are divisible by a prime p.

If a word in X and Y had error $O(|\epsilon|^k)$ then in this integer form the same word would be $I +$ (something divisible by p^k) and look for equality of matrices.

This does not give you proof that words work. This gives you a proof that they don't work. The surviving words are valid with high probability. The algorithm works in coRP, just like Miller-Rabin for primality. Change prime and random matrices and maybe at the end check with exact symbolic algebra to test if the remaining words are valid. This is based on the wisdom on polynomial identity testing (PIT).

4) Algebraic abstraction, the size of the matrix for X matters, but $SU(2)$ and $SL(2, \mathbb{F})$ are equivalent.

7 Lecture 7 (28 January 2022)

7.1 Self-correction degree in Kuperberg's modification of Kitaev-Shen-Vyalvi

Universal self-correction degree = Universal nilpotent degree. $\implies$ is easier.

Suppose that $X = I + \varepsilon A$ and $Y = I + \delta B$. Then, lemma $[X, Y] = 1 + O(\varepsilon\delta)$. Suppose that X and Y generate a free group F_2 and that $\mathbb{F}_2$ is linearly represented in some matrix group in this way.

Lower central series of F_2 is defined inductively,

$$(F_2)_n = \langle [(F_2)_{n-1}, F_2] \rangle$$

Using the lemma, if both X and Y are $I + O(\epsilon)$ then by induction any element W of $(F_2)_n$ is $I + O(\epsilon^n)$.

The other direction requires large matrices, and it requires an explicit model for the free $(n-1)$-step nilpotent group $(F_2)/(F_2)_n$ = a certain explicit linear representation where you can see that there is no other cancellation. That you can do this isn't obvious.

A related theorem: You can make $N_{2,n}$ or $N_{\text{anything},N}$ as a composition of $\mathbb{Z}$s. This is the "polycyclic construction". Then there exists a linear representation matching this decomposition, where all matrices of all elements of $N_{2,n}$ are unitriangular *and* each infinite cyclic factor lands in a different matrix entry.

As an example, $N_{2,2}$ is the integer Heisenberg group which is equal to

$$\begin{bmatrix} 1 & a & b \\ 0 & 1 & c \\ 0 & 1 & 1 \end{bmatrix}$$

a, b and c are integers. In another basis,

$$\begin{bmatrix} 1 & a\epsilon & b\epsilon^2 \\ 0 & 1 & c\epsilon \\ 0 & 0 & 1 \end{bmatrix}$$

is the same group, for fixed ϵ. If

$$X = \begin{bmatrix} 1 & \epsilon & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and

$$Y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \epsilon \\ 0 & 0 & 1 \end{bmatrix}$$

then error correction degree = nilpotent degree up to 2.

Pick some integer n. Then you can make some big matrices, bigger than $n \times n$, that has the same properties as this example, but upto ϵ^n than ϵ^2. Then for those matrices, the correction degree = nilpotent degree, upto the value n. This shows you that the universal correction degree is equal to the nilpotent degree and cannot be a larger integer. The hard part that we're skipping over is how to make these larger matrices. We said that those matrices are unitriangular but we didn't say that they're full unitriangular. The matrices get more and more sparse. They have a complicated pattern.

8 Lecture 8 (4 February 2022)

8.1 Modified Solovay-Kitaev algorithm

Toy model in $\mathbb{R}^n$: We want to reach (approximate) a target $\vec{v} \in \mathbb{R}^n$ (under some rules). We will approximate it backwards by subtracting off, at a low "cost".

Suppose $b_j^{(k)}$ is a well-conditioned infinite sequence of basis of $\mathbb{R}^n$. $\forall k, \vec{b}_1^{(k)}, \ldots, \vec{b}_n^{(k)}$ is a well-conditioned basis of $\mathbb{R}^n$. Well-conditioned means the bases (we're considering a sequence of basis sets) are roughly orthonormal and they shrink at a roughly regular rate. These are used for finer and finer cancellations.

$\forall k$, $\vec{b}_1^{(k)}, \ldots, \vec{b}_n^{(k)}$ is a well-conditioned basis – the matrix B is well-conditioned, condition number $\# \leq C$. A well-conditioned sequence has each $B^{(k)}$ is w.c. $||B^{(k)}||.||{B^{(k)}}^{-1}|| < C$. The condition number $\#$ is $||B^{(k)}||.||{B^{(k)}}^{-1}||$.

For a well-conditioned sequence each basis is w.c. and also the norms shrink at an approximately regular rate: $r||B^{(k-1)}|| > ||B^{(k)}|| > r^2||B^{(k-1)}||$ for some $0 < r < 1$. The choice of norm doesn't matter but the standard one is the L^2 operator norm, for some $r > 0$. It has to shrink but it can't shrink too much. Say that the cost of use of $\vec{b}_j^{(k)}$ is $O(k^\alpha)$. In other words, you have to purchase each use of the vector. Finer adjustments cost more. Anyway, then $\vec{V}$ can be approximated to within ϵ at cost $O(|\log \epsilon|^{\alpha+1})$. Proof: Shorten $\vec{v}$ with each $B^{(k)}$ in turn with a greedy algorithm.

The amount that you subtract off at each stage is bounded by r. So it's only a constant number of operations at each level. They come at this cost, which is not constant. But the total use of all the vectors at a given stage k is $O(k^\alpha)$. Then also that the vector has to shrink by something close to the scale of the basis. What happened to v after this bounded amount of work, v is bounded compared to the basis at the k-th stage.

Approximating any $\vec{v} \in \mathbb{R}^n$ can be reduced to a problem of shortening the error vector. So at each stage you try to subtract off the basis vectors to get the shortest length of error possible. Then in the next stage repeat the same process with finer basis vectors. And so on.

If the bases are good (like outright orthonormal) then it's clear will happen – it will correct each orthogonal direction separately. It will nevertheless work well enough as long as the lengths of the basis vectors aren't too disparate and the angles aren't too small. As long as they're orthonormal within some constant factor, you will be able to whittle down $\vec{v}$ to within some constant factor, as well as you could have done, being ready for the next stage.

You don't get to set the basis though; you have to order the basis sequence from Amazon! You can only apply the basis vectors you've been given. This is sufficient for the conclusion of the theorem; it's a more general hypothesis. The orientation of the basis vectors at each stage may change.

In a vector space, "frame" is another name for a basis. Frame elements are basis elements. We want to say frame in a group; it's not a basis in the algebraic sense. In the coordinate system of the Lie group near the identity it will look like a basis. It's still a basis in that sense. The same theorem holds in a Lie group with a sequence of frames or a well conditioned sequence of bases in its Lie algebra L.

The other part is making a well-conditioned sequence of bases for $SU(2)$ (say). It's easier to describe for $SL(2)$. It's easier to describe because the Lie bracket of the Lie algebra is just the cross product. So the idea is to make a well-conditioned sequence of 1st vectors components (or 1st legs) $g_1^{(k)}$. Then after that complete to frames $g_1^{(k)}, g_2^{(k)}, g^{(k)}$ from just the first one, and the original gate set Γ. Well-conditioned: $r||g^{(k-1)}|| > ||g^{(k)}|| > r^2||g^{(k-1)}||$. We're weakening the demand: just one vector at each scale. If we can make the other frame elements from one frame element, we're good.

Γ generates a dense subgroup of $SU(2)$ so $\exists$ a bounded-length words W_2, W_3 s.t. $g_1^{(k)}, g_2^{(k)} = w_2 g_1^{(k)} w_2^{-1}, g_3^{(k)} = w_3 g_1^{(k)} w_3^{-1}$ is well-conditioned. They are actually exactly the same norm because they are conjugates, and you can make them to within say 10% at a bounded extra cost. The extra fee now being the word length of the generators.

To make $g_1^{(k)}$ we only need $r^{k+1} < ||g_1^{(k)}|| < r^k$. So find some previous $r^{k/2-3} < ||g_1^{(l)}|| < r^{k/2-1}$ and now you can let the new vector be the group commutator of $g_1^{(k)} = [g_1^{(l)}, wg_1^{(l)}w^{-1}]$ where $wg_1^{(l)}w^{-1}$ is of bounded length and is chosen to adjust angle between x and y. The gate cost of $g_1^{(k)} = O(k^2)$. Total cost of approximation at a target x is $O(k^3)$. The group commutator is a non-commutative version of the cross product of vectors – it measures three things:
a) It's gonna be small if the rotation angles are small.
b) It's gonna be small if the angles between the axes are small.
c) " " " if the disparity between the lengths are small.

The gate cost of $g_1^{(k)}$ ends up being quadratic. This $g_1^{(k)} = [g_1^{(l)}, wg_1^{(l)}w^{-1}]$ is about four times as long a word from about halfway back. The key thing governing the exponent is that you get twice as far into the sequence where the word is four times as long. The scale of the frame gets squared so that the exponent roughly gets doubled for a word that's four times as long. If you think about what that means you get $O(k^2)$ by induction.

To use the entire well-conditioned sequence of frames the exponent gets bumped up by one as you have to use the frames at each level. The frames have to go in a roughly geometric progression; you're using the one that's about halfway back. In all versions of SK you go halfway back and take a commutator. That's always there.

The point is that the norm of $||g_1^{(k)}||$ is $O(||g_1^{(l)}||^2)$. That is, you get a corrector that's twice as far in the log scale for a cost of four times as much. SK had this idea in their original paper, but what people usually do not have is the well-conditioned sequence of frames. Kitaev-Shen-Vyalvi use a cloud or ϵ-net in $\mathbb{R}^n$. So they had a trouble getting exactly 2 in the exponent.

This was twice as far in the ladder at four times the cost. This is where the new words come in. First thing to be learned from Elkasapy and Thom is that we can go five times as far at fourteen times the cost. The exponent changes to $\log_5(14)$ (not even 2!) and then this is whatever the output is.

So $X = I + O(\epsilon)$ and $Y = I + O(\epsilon)$ and $[X, Y] = XYX^{-1}Y^{-1} = I + O(\epsilon^2)$. A priori, the commutator of $[[X, Y], [Y, Z]] = I + O(\epsilon^4)$. But $[[X, Y], [Y, X^{-1}]] = I + O(\epsilon^5)$, which is a happy accident!

So $g_1^{(k)} = w(g_1^{(l)}, xg_1^{(l)}x^{-1})$ where x is of bounded length. The norm of the new one $||g_1^{(k)}|| = ||g_1^{(l)}||^t f(\theta) + O(||g_1^{(l)}||^t)$ where $f(\theta)$ was $\sin(\theta)$ becomes a more complicated function. The precept that you only need to get within bounded precision of x stays. This angle between two group elements is troublesome for larger matrices like those in $SU(3)$. At least upto first order, the $f(\theta)$ is equivalent to the angle between the two Boolean measurements of a qubit. Then there's this fact: if you have a quantum measurement for a qubit, they just have an angle in the Bloch sphere. Suppose now you have two complete measurements for a qutrit: they have four angles – they're much trickier and messier. The relative position of two orthonormal line bases of a qutrit are given by four parameters. The parameters also become a mess – they're clean only in the qubit case.

The best words that we know of come from Elkasapy and Thom. As we discussed, the nilpotent degree matches the self-cancelling degree in the limit of bigger matrices. Those words are fine for any size of matrix – but they only have a conjecture that their words are optimal. And we also don't understand what their words do. We believe their exponents or nilpotent degree exponents. What these angle dependences will look like for their words, particularly for larger matrices, is unknown/hard. The trig polynomial might also have some interesting symmetry behaviour under group commutation.

9 Lecture 9 (11 February 2022)

9.1 Review of elementary circuit complexity concepts

qprob is the finite dimensional VNAs and TPCP maps on states is our official $\otimes$ category for quantum computation.
1. Uniform families of circuits.
2. Periodic circuits = TPCP cellular automata.
3. Stochastic TM with quantum tape (adaptive quantum circuit).

Any $\otimes$ category needs a gate set. You can exactly generate everything using all 2-qubit unitary gates $PU(4)$ and $0-1$ ary ancilla creation and qubit discard (and tacitly, qubit to bit measurement).

Usually you want a finite gate set that densely generates a priori with no rate of convergence. We need a SK theorem to get a theory of gate set independence. This has convergence with poly($|\log \epsilon|$) overhead. Normally poly($\frac{1}{\epsilon}$) would be good enough to define BQP.

For both BQP and BPP "polynomial time" is self-consistent ($\mathrm{BPP}^{\mathrm{BPP}} = \mathrm{BPP}$ and $\mathrm{BQP}^{\mathrm{BQP}} = \mathrm{BQP}$) w.r.t subroutine calls if it means poly($x, \frac{1}{\epsilon}$). poly($x, \log|\epsilon|$) is even better for gate independence theorem.

For prob/BPP it's true that AND, NOT, OR, COPY and COIN are universal.

Technical Fact: Some coin flips (and quantum gates) are too powerful a priori. So there is a rule that the digits of any gate in prob or qprob are computable in polytime or ideally in $\tilde{O}(n)$ time on a RAM machine.

Full picture of BQP, etc. uses TPCPs but it has unitary subcategory which is useful/necessary in particular algorithms.

9.2 Introduction to Shor's algorithm

Given a hiding function $f\colon \mathbb{Z} \to X$ where f is some classical thing and X is a symbol set which is:
1. Periodic $f(x+h) = f(x)$
2. Otherwise aperiodic $f(x) \neq f(y)$ if $h \nmid x-y$

Given an algorithm ("functional input") for f, Shor can find h in time poly($\log h$). Factoring integers is one small special case.

What Shor really needs to make f is a unitary embedding $|x\rangle \mapsto |x, f(x)\rangle$. f must be returned with no scratchwork or no hidden measurement. See Why do we have to uncompute rather than simply set registers to zero?.

Reversible computing is the classical counterpart to unitary. Reversible computing has permutation as gates.

Tofolli gate: $\{a,b,c\} \mapsto \{a,b,ab+c\}$. This is a permutation of 8-bit strings of length 3. But for a complete picture the category is is injective partial functions (can throw exceptions), not just bijections. With this convention, any circuit in P can be dilated to RevP even without leaving scratchwork!

1st draft with scratchwork left

NOT - Okay.

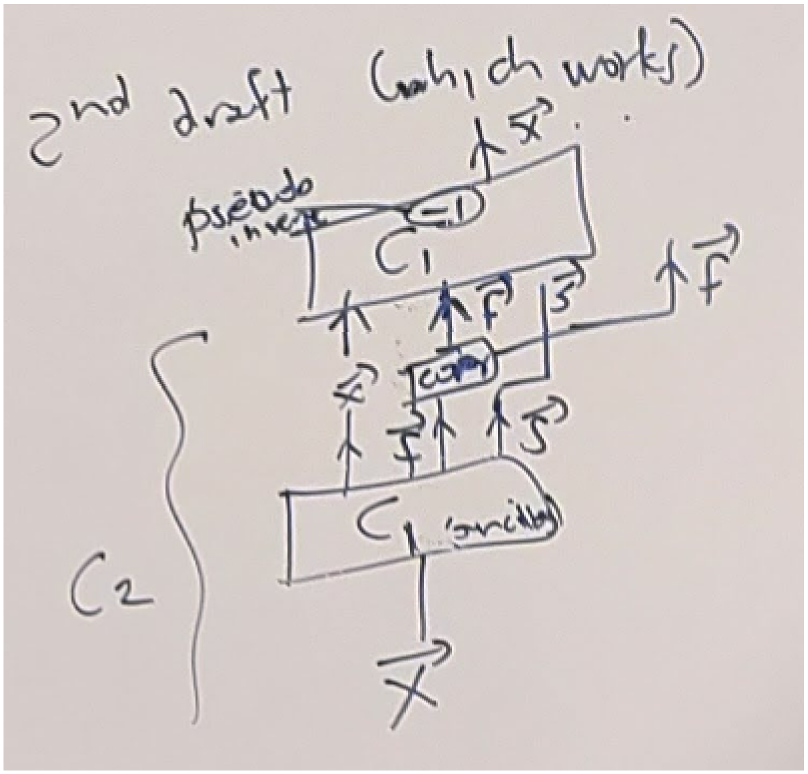
Change AND to $\{a,b,0\} \mapsto \{a,b,ab+0\}$. The $\{a,b\}$ in the output is scratchwork.

CNOT: $\{a,0\} \mapsto \{a,a\} = \mathrm{COPY}$.

No version of discard is reversible even by this expanded standard. Say, you discard a, b in Toffoli gate in the output it would be a map from a 4-point set to a 2-point set. (Though you can discard a bit if you "know" its value.)

Goal: $|x\rangle \mapsto |x, f(x)\rangle$ or $x \mapsto x, f(x)$.

2nd draft which works



Any partial injection is a pseudoinverse because you can just map back each point to the domain set (inverse gives blank on points outside the domain that throw exception). We can do a limited amount of copy, namely $|0\rangle \mapsto |00\rangle$ and $|1\rangle \mapsto |11\rangle$ but not $|\psi\rangle \mapsto |\psi\psi\rangle$.

The reversible classical category embeds in the subunitary category STPCP (not very different from TPCP):

$$||Ux||_2 \leq ||x||_2$$

and

$$\langle\psi|U^\dagger U|\psi\rangle \leq \langle\psi, \psi\rangle$$

A bit more of Shor:

1. Choose $q > h^2$ where q is the number of qubits and h as in $f(x) = f(x+h)$. The idea is that there's a Gauche embedding of $\mathbb{Z}/2^q \hookrightarrow \mathbb{Z}$.
2. Make the equal superposition on q qubits.

$$|+\rangle^q = \frac{1}{2^{q/2}} \sum_{x\in\mathbb{Z}/2^q} |x\rangle = |\psi\rangle.$$

3. Evaluate $D_f|\psi\rangle$ to get

$$\frac{1}{2^{q/2}} \sum_{x\in\mathbb{Z}/2^q} |x, f(x)\rangle.$$

Throw away the output and analyze the partially measured input. The measurement is not to help the algorithm but to analyze it. You would measurement those qubits to keep the partially measured qubits in a pure state. It's not the algorithm that measures the $f(x)$ qubits at all; it's you who is measuring to analyze the algorithm. (You can either model it as a hidden measurement or a visible measurement with a posterior state.)

$$|0\rangle \mapsto |0, \text{apple}\rangle$$
$$|1\rangle \mapsto |1, \text{banana}\rangle$$
$$|2\rangle \mapsto |2, \text{apple}\rangle$$
$$|3\rangle \mapsto |3, \text{banana}\rangle$$

If you throw away the apple/banana qubits, in effect, the environment partially measured the x qubits by transferring them to their posterior state.

If you measure the fruit part only:

$$a_0|0\rangle + a_1|1\rangle + a_2|2\rangle + a_3|3\rangle \mapsto (a_0|0\rangle + a_2|2\rangle) \otimes |\text{apple}\rangle$$
$$a_0|0\rangle + a_1|1\rangle + a_2|2\rangle + a_3|3\rangle \mapsto (a_1|0\rangle + a_3|2\rangle) \otimes |\text{banana}\rangle$$

The h^2 comes from the later Fourier stage which results in a rational number with denominator h. So then Shor's algorithm has classical post-processing to determine h from this rational number. But this rational number is not learned exactly; it's only learned in decimal form upto q binary bits of precision. You get a number of the form r/h and you want to learn the denominator.

How many digits do you want to find the numerator and denominator? If 2 digits each, then 4 digits of precision at least required. How close can r/h be to r'/h'? Well, if you cross-multiply you'd get $|(rh' - r'h)/hh'| \geq 1/hh'$. This is where the square comes in.