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

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

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

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 $\rho$ vs. $\sigma$ or $E$ vs. $F$ for any use with only one copy.

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

Worst case infidelity,

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

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

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

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

$\mathsf{P}/\mathsf{poly}$ 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:** $\mathsf{P} = \mathsf{P}_{\text{uniform}}$

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

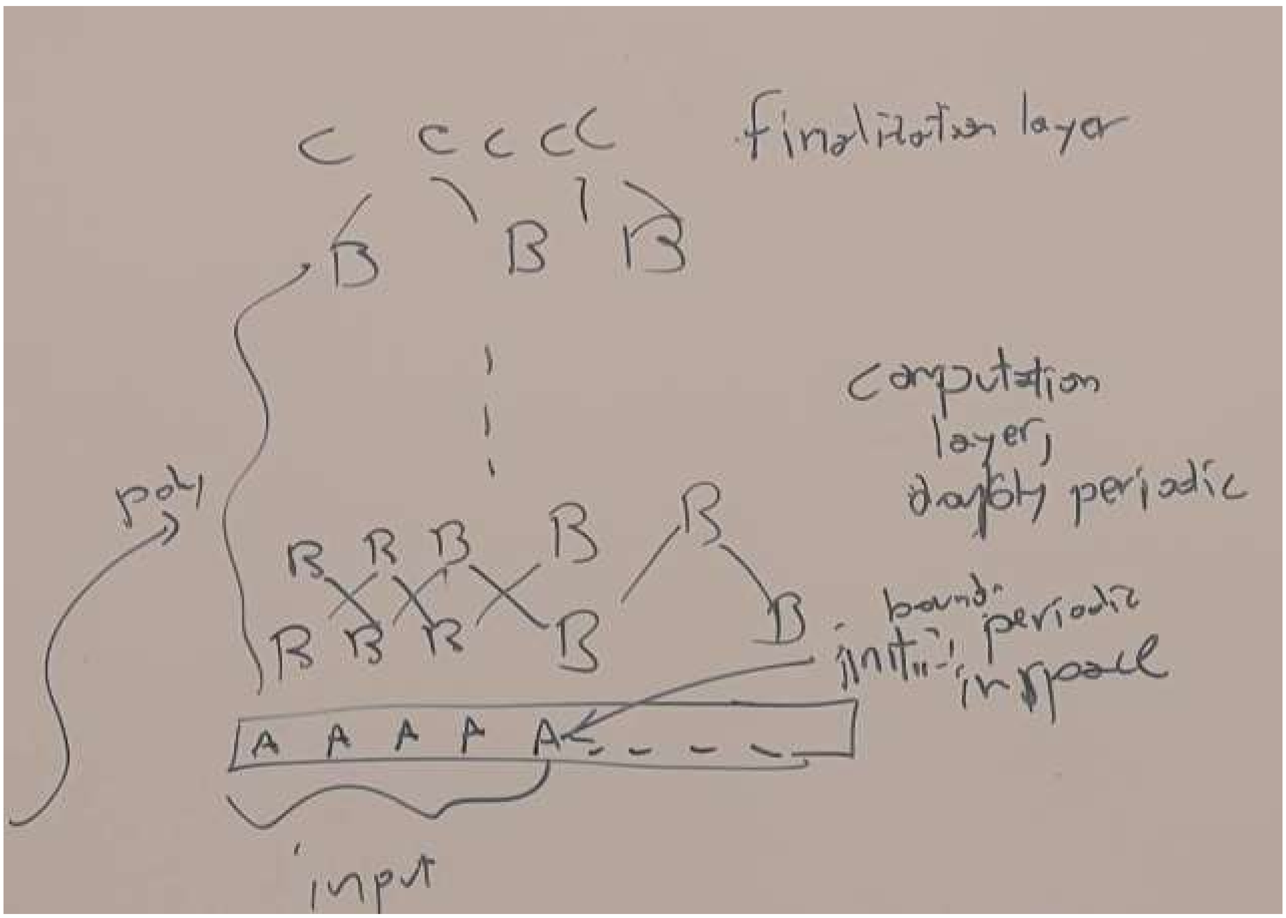
The $\supseteq$ containment is thought of as simulating your own circuit. The $\subseteq$ containment is an unrolling argument.

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




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 $(\mathbb{Z}/2\mathbb{Z})^n$.

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. (**Thm**: 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 $\mathrm{polylog}(\epsilon)$ overhead. This is basically the statement of the **Solovay-Kitaev theorem** in the quantum case.

Illustration of how computation is done by an automaton: