Reading, Analyzing, and Simulating Quantum Circuits

(With speculation on the status of "quantum supremacy")

Kenneth W. Regan[1]
University at Buffalo (SUNY)

RIT, 24 Apr., 2025

[1]Includes joint work with Amlan Chakrabarti, University of Calcutta, and Chaowen Guan, University of Cincinnati

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...
- The classical Simulation Hypothesis presumes it.

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...
- The classical Simulation Hypothesis presumes it.

Con:

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...
- The classical Simulation Hypothesis presumes it.

Con:

- "Nature isn't classical, dammit!" (Richard Feynman, whole quote)

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...
- The classical Simulation Hypothesis presumes it.

Con:

- "Nature isn't classical, dammit!" (Richard Feynman, whole quote)
- Experience with exponential blowups in simulations.

Is Nature Lexical?

I.e., can all natural processes be simulated in proportional time by computers using today's programming languages?

Pro:

- "It From Bit." "Unreasonable Effectiveness of Mathematics."
- *Church-Turing Thesis* extended to physics and **feasible** computation (as formalized e.g. by the polynomial-time class P).
- Stephen Wolfram's cellular automata universe; others' models...
- The classical Simulation Hypothesis presumes it.

Con:

- "Nature isn't classical, dammit!" (Richard Feynman, whole quote)
- Experience with exponential blowups in simulations.
- "It From Qubit."

The Reach of Mathematics

The Reach of Mathematics

- Can **define** **and analyze** entities that cannot be computed.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
    - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
    - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
    - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
    - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.
- For other entities we strongly doubt feasibility:

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.
- For other entities we strongly doubt feasibility:
  - $Q = \{$true sentences using only $(\wedge, \vee, \neg, \exists, \forall)\}$. (PSPACE-complete)

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.
- For other entities we strongly doubt feasibility:
  - $Q = \{$true sentences using only $(\wedge, \vee, \neg, \exists, \forall)\}$. (PSPACE-complete)
  - SAT $= \{$true sentences using only $(\wedge, \vee, \neg, \exists)$. (NP-complete)

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.
- For other entities we strongly doubt feasibility:
  - $Q = \{$true sentences using only $(\wedge, \vee, \neg, \exists, \forall)\}$. (PSPACE-complete)
  - SAT $= \{$true sentences using only $(\wedge, \vee, \neg, \exists)$. (NP-complete)
  - We know $\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE}$ but have not proved $\mathsf{P} \neq \mathsf{PSPACE}$.

The Reach of Mathematics

- Can **define and analyze** entities that cannot be computed.
  - $V = \{$true statements about integers in formal arithmetic (PA)$\}$.
  - Tarski, 1923: definable in set theory but not within PA itself.
- Can compute entities in some cases but not others.
  - $A = \{$provable statements of PA$\}$. (Gödel, Turing, Church, 1930s)
- Can compute other entities but **not in feasible time**.
  - $B = \{$true statements of PA using $+, =$ but not $\cdot\}$.
  - Decidable by Presburger, 1929; infeasible by Fischer and Rabin, 1974.
- For other entities we strongly doubt feasibility:
  - $Q = \{$true sentences using only $(\wedge, \vee, \neg, \exists, \forall)\}$. (PSPACE-complete)
  - SAT $= \{$true sentences using only $(\wedge, \vee, \neg, \exists)$. (NP-complete)
  - We know $\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE}$ but have not proved $\mathsf{P} \neq \mathsf{PSPACE}$.
- So Math can outpace its own calculations, but can Nature?

Concreteness and Complexity

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
    - Perfect chess has been **tablebased** for up to 7 pieces on the board.
    - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6, 6)$? (Erdős quote)
    - We know $102 \leq R(6, 6) \leq 162$.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut?

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??
- **Factoring** m-bit numbers seems concretely hard **in most cases**.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??
- **Factoring** m-bit numbers seems concretely hard **in most cases**.
- Belief in asymptotic classical time lower bound $2^{\tilde{\Omega}(m^{1/3})}$.

Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??
- **Factoring** m-bit numbers seems concretely hard **in most cases**.
- Belief in asymptotic classical time lower bound $2^{\tilde{\Omega}(m^{1/3})}$.
  - But no wide-ranging *hardness* result. (Maybe exponent is 1/4? 1/5?)

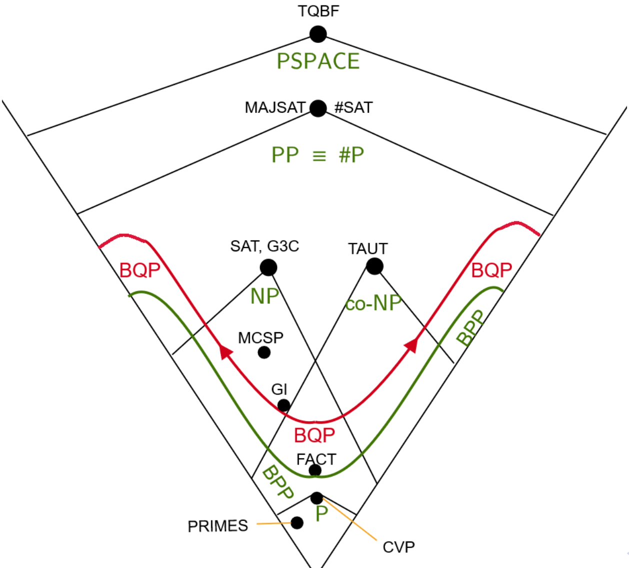
Concreteness and Complexity

- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??
- **Factoring** m-bit numbers seems concretely hard **in most cases**.
- Belief in asymptotic classical time lower bound $2^{\tilde{\Omega}(m^{1/3})}$.
  - But no wide-ranging *hardness* result. (Maybe exponent is 1/4? 1/5?)
- Hence a shock in 1993-94 when Peter Shor put factoring into **BQP**.

Concreteness and Complexity

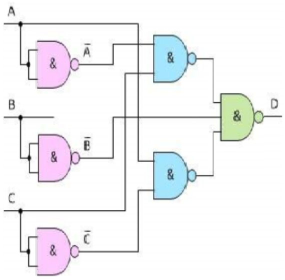
- Perfect chess and Go on $n \times n$ boards are PSPACE-complete (under extensions of common rules limiting the length of games).
- On concrete 8×8 (respectively, 19×19)boards, **perfection** remains a challenge.
  - Perfect chess has been **tablebased** for up to 7 pieces on the board.
  - A laptop holding 32-piece tables would collapse to a black hole.
- Will we ever compute the Ramsey number $R(6,6)$? (Erdős quote)
  - We know $102 \leq R(6,6) \leq 162$.
  - $R(5,5)$ still open, recently tightened to $43 \leq R(5,5) \leq 46$.
  - Is there a cosmic shortcut? Maybe if $\mathsf{NP} = \mathsf{P}$ super-concretely??
- **Factoring** m-bit numbers seems concretely hard **in most cases**.
- Belief in asymptotic classical time lower bound $2^{\tilde{\Omega}(m^{1/3})}$.
  - But no wide-ranging *hardness* result. (Maybe exponent is 1/4? 1/5?)
- Hence a shock in 1993-94 when Peter Shor put factoring into **BQP**.
- Realizable by **quantum circuits** of size $\tilde{O}(m^2)$.

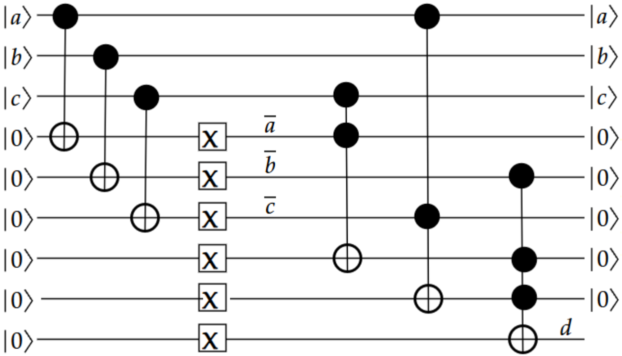
The Complexity Class Neighborhood



Classical and Quantum Circuits

$d = f(a,b,c) = \bar{\wedge}\,(c \;\bar{\wedge}\; \bar{a}, \bar{b}, a \;\bar{\wedge}\; \bar{c})$ $\quad$ $\mathbf{X}$ = NOT, ●—⊕ = controlled-NOT





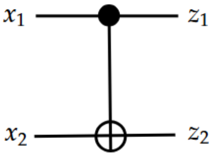
Quantum circuit computes the **reversible** Boolean function
$F(a,b,c,e_1,e_2,e_3,e_4,e_5,e_6) = (a,b,c,e_1,e_2,e_3,e_4,e_5,e_6 \oplus f(a,b,c))$.
Underlying: a vector of $N = 2^9 = 512$ dimensions.

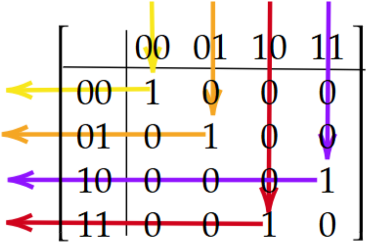
Quantum Coordinates and the CNOT Gate

By **linearity**, an n-qubit circuit is determined by its actions on the 0-1 **standard basis** vectors $e_{0^n} = [1,0,0,\dots,0]^T$ thru $e_{1^n} = [0,0,0,\dots,1]^T$.

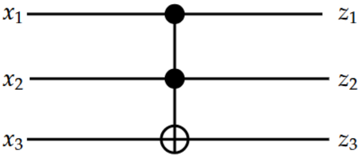
For $n=2$, $e_{00} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $e_{01} = \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $e_{10} = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$; and $e_{11} = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$ (in "big-endian" order). Then **CNOT**$e_{00} = e_{00}$ and **CNOT**$e_{01} = e_{01}$, while **CNOT**$e_{10} = e_{11}$ and **CNOT**$e_{11} = e_{10}$. **Feynman path** visualization:

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$





Toffoli Gate ($n = 3,\ N = 2^n = 8$)



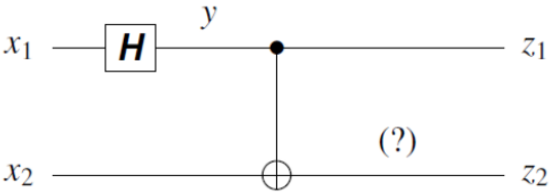
$$
\left[\begin{array}{c|cccccccc}
 & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\
\hline
000 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
001 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
010 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
011 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
100 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
101 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
110 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
111 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}\right]
$$

In **Dirac notation**, e_x is written $|x\rangle$. So we have **Tof** $|000\rangle = |000\rangle$ thru **Tof** $|101\rangle = |101\rangle$, while **Tof** $|110\rangle = |111\rangle$ and **Tof** $|111\rangle = |110\rangle$.

Note that fixing $x_3 = 1$ makes $z_3 = x_1 \barwedge x_2$. Since NAND is a universal gate, this already suffices to show that quantum circuits simulate classical ones. Their extra power comes from one more gate.

Hadamard Gate, Nondeterminism, and Entanglement

Hadamard gate H $= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ "emits" a free Boolean variable y.



On input $|00\rangle$, $y = 0$ leads to output $z_1 = 0$, $z_2 = 0$, while $y = 1$ leads to $z_1 z_2 = 11$. The other combinations 01 and 10 cannot happen. *Matrices:*

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Measurement and Sampling

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.
- **Unitary** matrices A, meaning $AA^* = I$, preserve unit vectors.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.
- **Unitary** matrices A, meaning $AA^* = I$, preserve unit vectors.
- Measuring the **entangled** state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ returns $|00\rangle$ with probability 0.5 or $|11\rangle$ with probability 0.5, but never $|01\rangle$ or $|10\rangle$.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.
- **Unitary** matrices A, meaning $AA^* = I$, preserve unit vectors.
- Measuring the **entangled** state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ returns $|00\rangle$ with probability 0.5 or $|11\rangle$ with probability 0.5, but never $|01\rangle$ or $|10\rangle$.
- So an n-qubit circuit C, on any input $|x\rangle$, gives rise to a distribution $D_{C,x}$ on $\{0,1\}^n$ to sample from.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \dots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.
- **Unitary** matrices A, meaning $AA^* = I$, preserve unit vectors.
- Measuring the **entangled** state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ returns $|00\rangle$ with probability 0.5 or $|11\rangle$ with probability 0.5, but never $|01\rangle$ or $|10\rangle$.
- So an n-qubit circuit C, on any input $|x\rangle$, gives rise to a distribution $D_{C,x}$ on $\{0,1\}^n$ to sample from.
- Google's *quantum supremacy* methodology creates a family of C such that non-negligible values of $D_{C,\_}$ are hard to find classically.

Measurement and Sampling

- **Measuring** the state vector $|\phi\rangle = [a_0, a_1, \ldots, a_{N-1}]^T$ of all n qubits returns some i, $0 \leq i \leq N-1$, with probability $|a_i|^2$. You can say it returns e_i, $|i\rangle$, or the i-th binary string $z_i \in \{0,1\}^n$.
- This entails that $|\phi\rangle$ is a Euclidean **unit vector**, so that the probabilities sum to 1.
- **Unitary** matrices A, meaning $AA^* = I$, preserve unit vectors.
- Measuring the **entangled** state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ returns $|00\rangle$ with probability 0.5 or $|11\rangle$ with probability 0.5, but never $|01\rangle$ or $|10\rangle$.
- So an n-qubit circuit C, on any input $|x\rangle$, gives rise to a distribution $D_{C,x}$ on $\{0,1\}^n$ to sample from.
- Google's *quantum supremacy* methodology creates a family of C such that non-negligible values of $D_{C,\_}$ are hard to find classically.
- **Shor's algorithm** creates $D_{C,x}$ such that sampling often gives z from which the period r of $f_a(u) = a^u \bmod x$ can be classically inferred, which in turn often enables factoring x.

Some More Gates

$$\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathsf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathsf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad \mathsf{R_8} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix},$$

$$\mathsf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad \mathsf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

Some More Gates

$$\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathsf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathsf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad \mathsf{R}_8 = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix},$$

$$\mathsf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad \mathsf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

- The gates $\mathsf{H}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{S}, \mathsf{CNOT}, \mathsf{CZ}$ generate *Clifford circuits*, which are simulatable in polynomial time.

Some More Gates

$$\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathsf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathsf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad \mathsf{R}_8 = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix},$$

$$\mathsf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad \mathsf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

- The gates $\mathsf{H}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{S}, \mathsf{CNOT}, \mathsf{CZ}$ generate *Clifford circuits*, which are simulatable in polynomial time. **(Time improved by us.)**

Some More Gates

$$\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathsf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathsf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad \mathsf{R_8} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix},$$

$$\mathsf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad \mathsf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

- The gates $\mathsf{H}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{S}, \mathsf{CNOT}, \mathsf{CZ}$ generate *Clifford circuits*, which are simulatable in polynomial time. **(Time improved by us.)**
- Adding any of T, $\mathsf{R_8}$, CS, or Tof gives the full power of BQP.

Some More Gates

$$\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathsf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

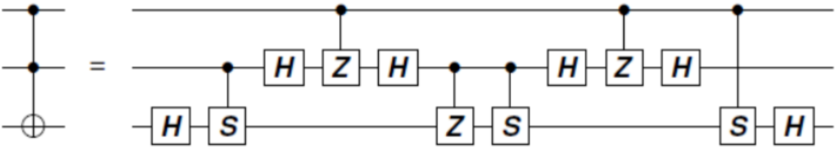
$$\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathsf{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad \mathsf{R_8} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix},$$

$$\mathsf{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \quad \mathsf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

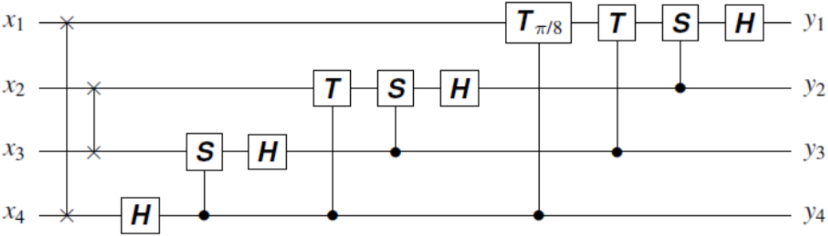
- The gates $\mathsf{H}, \mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{S}, \mathsf{CNOT}, \mathsf{CZ}$ generate *Clifford circuits*, which are simulatable in polynomial time. **(Time improved by us.)**
- Adding any of T, $\mathsf{R_8}$, CS, or Tof gives the full power of BQP.
- Note: $\mathsf{T}^2 = \mathsf{S}$, $\mathsf{S}^2 = \mathsf{Z}$, $\mathsf{Z}^2 = \mathsf{I} = \mathsf{H}^2$, and $\mathsf{CS}^2 = \mathsf{CZ}$.

Two Notable Circuits

H and **CS** alone can simulate the Toffoli gate:

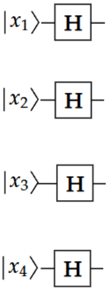


The 4-qubit **Quantum Fourier Transform** == the 16×16 **Discrete Fourier Transform**. In general, $\mathbf{QFT}_n$ needs $O(n^2)$ basic gates.



Juxtaposition and Tensor Product

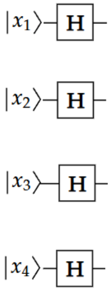
Many QCs begin with m Hadamard gates on each of m qubits



$$\frac{1}{4}\left[\begin{array}{c|cccccccccccccccc}
\mathbf{H}^{\otimes 4} & 0000 & 0001 & 0010 & 0011 & 0100 & 0101 & 0110 & 0111 & 1000 & 1001 & 1010 & 1011 & 1100 & 1101 & 1110 & 1111 \\
\hline
0000 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0001 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
0010 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
0011 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 \\
0100 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
0101 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
0110 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
0111 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\
1000 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1001 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
1010 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\
1011 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\
1100 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\
1101 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 \\
1110 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 \\
1111 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1
\end{array}\right]$$

Juxtaposition and Tensor Product

Many QCs begin with m Hadamard gates on each of m qubits



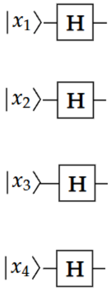
$\frac{1}{4}$

$\mathbf{H}^{\otimes 4}$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0001	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1
0010	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1
0011	1	−1	−1	1	1	−1	1	−1	1	−1	−1	1	1	−1	1	−1
0100	1	1	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1
0101	1	−1	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1
0110	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1	1	1
0111	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1	1	−1
1000	1	1	1	1	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1
1001	1	−1	1	−1	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1
1010	1	1	−1	−1	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1
1011	1	−1	−1	1	1	−1	1	−1	−1	1	1	−1	−1	1	1	−1
1100	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1	1	1	1	1
1101	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1	1	−1	1	−1
1110	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1	1	1	−1	−1
1111	1	−1	−1	1	−1	1	1	−1	−1	1	1	−1	1	−1	−1	1

- Is this “4” units of work? Or “16”? Or “256”?

Juxtaposition and Tensor Product

Many QCs begin with m Hadamard gates on each of m qubits



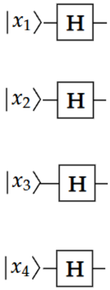
$\frac{1}{4}$

$\mathbf{H}^{\otimes 4}$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0001	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1
0010	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1
0011	1	−1	−1	1	1	−1	1	−1	1	−1	−1	1	1	−1	1	−1
0100	1	1	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1
0101	1	−1	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1
0110	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1	1	1
0111	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1	1	−1
1000	1	1	1	1	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1
1001	1	−1	1	−1	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1
1010	1	1	−1	−1	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1
1011	1	−1	−1	1	1	−1	1	−1	−1	1	1	−1	−1	1	1	−1
1100	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1	1	1	1	1
1101	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1	1	−1	1	−1
1110	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1	1	1	−1	−1
1111	1	−1	−1	1	−1	1	1	−1	−1	1	1	−1	1	−1	−1	1

- Is this “4” units of work? Or “16”? Or “256”?
- On one hand, the rule $\mathbf{H}^{\otimes n}[u, v] = \frac{1}{\sqrt{n}}(-1)^{u \bullet v}$ for entries is simple.

Juxtaposition and Tensor Product

Many QCs begin with m Hadamard gates on each of m qubits



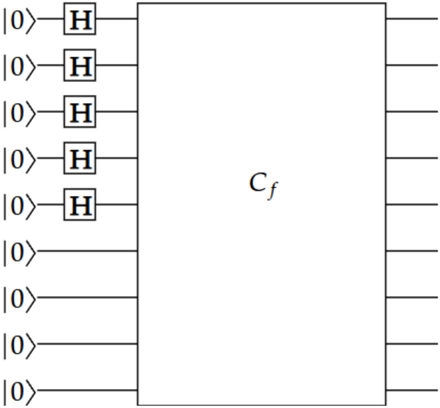
$\frac{1}{4}$

$\mathbf{H}^{\otimes 4}$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0001	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1	−1
0010	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1	1	1	−1	−1
0011	1	−1	−1	1	1	−1	1	−1	1	−1	−1	1	1	−1	1	−1
0100	1	1	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1
0101	1	−1	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1
0110	1	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1	−1	1	1
0111	1	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1	1	1	−1
1000	1	1	1	1	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1
1001	1	−1	1	−1	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1
1010	1	1	−1	−1	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1
1011	1	−1	−1	1	1	−1	1	−1	−1	1	1	−1	−1	1	1	−1
1100	1	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1	1	1	1	1
1101	1	−1	1	−1	−1	1	−1	1	−1	1	−1	1	1	−1	1	−1
1110	1	1	−1	−1	−1	−1	1	1	−1	−1	1	1	1	1	−1	−1
1111	1	−1	−1	1	−1	1	1	−1	−1	1	1	−1	1	−1	−1	1

- Is this "4" units of work? Or "16"? Or "256"?
- On one hand, the rule $\mathbf{H}^{\otimes n}[u,v] = \frac{1}{\sqrt{n}}(-1)^{u \bullet v}$ for entries is simple.
- On the other, some claim this involves splitting off 2^n branches of a multiverse.

Hadamard Transforms and Functions

Suppose $C_f(x, y) = (x, y \oplus f(x))$ computes the reversible form of f. Then

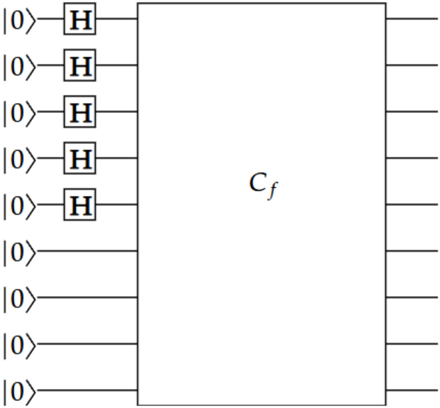


computes what I call the **functional superposition**

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |f(x)\rangle.$$

Hadamard Transforms and Functions

Suppose $C_f(x, y) = (x, y \oplus f(x))$ computes the reversible form of f. Then



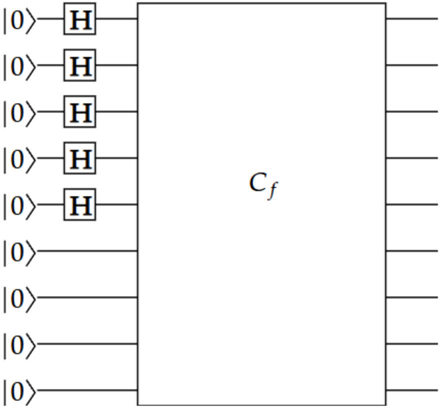
computes what I call the **functional superposition**

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |f(x)\rangle.$$

- Maybe this has exponentially many “jaggies”?

Hadamard Transforms and Functions

Suppose $C_f(x, y) = (x, y \oplus f(x))$ computes the reversible form of f. Then



computes what I call the **functional superposition**

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |f(x)\rangle.$$

- Maybe this has exponentially many "jaggies"?
- *We've now seen all the ingredients of Shor's Algorithm.*

Three Universal Libraries, Phase Angles, and Noise

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)
- Programmed improvement by Peter Selinger and Neil Ross.

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)
- Programmed improvement by Peter Selinger and Neil Ross.
- The gate set $\mathsf{H} + \mathsf{Tof}$ is not metrically universal—it has no complex scalars—but it is **computationally universal**: It can maintain real and complex parts of quantum states in double-rail manner.

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)
- Programmed improvement by Peter Selinger and Neil Ross.
- The gate set $\mathsf{H} + \mathsf{Tof}$ is not metrically universal—it has no complex scalars—but it is **computationally universal**: It can maintain real and complex parts of quantum states in double-rail manner.
- The gate set $\mathsf{H} + \mathsf{CS}$ is efficiently metrically universal.

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)
- Programmed improvement by Peter Selinger and Neil Ross.
- The gate set $\mathsf{H} + \mathsf{Tof}$ is not metrically universal—it has no complex scalars—but it is **computationally universal**: It can maintain real and complex parts of quantum states in double-rail manner.
- The gate set $\mathsf{H} + \mathsf{CS}$ is efficiently metrically universal.
- Thus we don't need arbitrarily fine-angled *gates* to compute $\mathbf{QFT}_n$ finely enough with $\tilde{O}(n^2)$ basic gates.

Three Universal Libraries, Phase Angles, and Noise

- The gate set $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ is **efficiently metrically universal**, meaning that any feasible quantum circuit of size s can be approximated to within entrywise error ϵ by a circuit of these gates only in size $O(s) \cdot (\log \frac{s}{\epsilon})^{O(1)}$. (See Solovay-Kitaev theorem.)
- Programmed improvement by Peter Selinger and Neil Ross.
- The gate set $\mathsf{H} + \mathsf{Tof}$ is not metrically universal—it has no complex scalars—but it is **computationally universal**: It can maintain real and complex parts of quantum states in double-rail manner.
- The gate set $\mathsf{H} + \mathsf{CS}$ is efficiently metrically universal.
- Thus we don't need arbitrarily fine-angled *gates* to compute $\mathbf{QFT}_n$ finely enough with $\tilde{O}(n^2)$ basic gates.
- But fine angles exist in the output and may be especially vulnerable to noise.

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, ..., x_n; \mathbf{y_1}, ..., \mathbf{y_h}; z_1, ..., z_n)$.

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n)$.
- An additive polynomial $q_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n; w_1, \ldots)$.

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n)$.
- An additive polynomial $q_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n; w_1, \ldots)$.
- A Boolean representation ϕ_C with various auxiliary variables.

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n)$.
- An additive polynomial $q_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n; w_1, \dots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C?

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, ..., x_n; \mathbf{y_1}, ..., \mathbf{y_h}; z_1, ..., z_n)$.
- An additive polynomial $q_C(x_1, ..., x_n; \mathbf{y_1}, ..., \mathbf{y_h}; z_1, ..., z_n; w_1, ...)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C?

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n)$.
- An additive polynomial $q_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n; w_1, \dots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C? **They perform poorly.**

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n)$.
- An additive polynomial $q_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n; w_1, \dots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C? **They perform poorly.**

3. Simplify intermediate states via logic?

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n)$.
- An additive polynomial $q_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n; w_1, \dots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C? **They perform poorly.**

3. Simplify intermediate states via logic? **Not promising so far...**

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n)$.
- An additive polynomial $q_C(x_1, \ldots, x_n; \mathbf{y_1}, \ldots, \mathbf{y_h}; z_1, \ldots, z_n; w_1, \ldots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C? **They perform poorly.**

3. Simplify intermediate states via logic? **Not promising so far...**

4. Maybe program **non-physical** approximations? **Hmmmmm...**

Demo of Simulation Code and Its Blue-Sky Ideas

Theorem (Regan-Chakrabarti-Guan): Given an n-qubit circuit C with h nondeterministic (Hadamard) gates, we can efficiently compute [...]:

- A product polynomial $p_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n)$.
- An additive polynomial $q_C(x_1, \dots, x_n; \mathbf{y_1}, \dots, \mathbf{y_h}; z_1, \dots, z_n; w_1, \dots)$.
- A Boolean representation ϕ_C with various auxiliary variables. **Used by the simulator.**

Idea 1: Bounds from algebraic-geometric invariants of ∂p_C? **Hard...**

2. Can **SAT solvers**—or really #SAT counters—heuristically evaluate C? **They perform poorly.**

3. Simplify intermediate states via logic? **Not promising so far...**

4. Maybe program **non-physical** approximations? **Hmmmmm...**

5. Iteratively program **tensor network and SVD approximations**... **On the agenda...**

I. Feynman Path Polynomials and Logical Formulas

Let C have "minphase" $K = 2^k$ and let F embed K-th roots of unity ω.

- $\mathsf{H} + \mathsf{Tof}$ has $k = 1$, $K = 2$.
- $\mathsf{H} + \mathsf{CS}$ has $k = 2$, $K = 4$.
- $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ has $k = 3$, $K = 8$.

I. Feynman Path Polynomials and Logical Formulas

Let C have "minphase" $K = 2^k$ and let F embed K-th roots of unity ω.

- $\mathsf{H} + \mathsf{Tof}$ has $k = 1$, $K = 2$.
- $\mathsf{H} + \mathsf{CS}$ has $k = 2$, $K = 4$.
- $\mathsf{H} + \mathsf{CNOT} + \mathsf{T}$ has $k = 3$, $K = 8$.

Theorem (RC 2007-09, extending Dawson et al. (2004) over $\mathbb{Z}_2$)

Any QC C of n qubits quickly transforms into a polynomial $P_C = \prod_g P_g$ over gates g and a constant $R > 0$ such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j (\#y : P_C(x,y,z) = \iota(\omega^j))$$

I. Feynman Path Polynomials and Logical Formulas

Let C have "minphase" $K = 2^k$ and let F embed K-th roots of unity ω.

- H + Tof has $k = 1$, $K = 2$.
- H + CS has $k = 2$, $K = 4$.
- H + CNOT + T has $k = 3$, $K = 8$.

Theorem (RC 2007-09, extending Dawson et al. (2004) over $\mathbb{Z}_2$)

Any QC C of n qubits quickly transforms into a polynomial $P_C = \prod_g P_g$ over gates g and a constant $R > 0$ such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j (\#y : P_C(x,y,z) = \iota(\omega^j)) = \frac{1}{R} \sum_y \omega^{P_C(x,y,z)},$$

where C has h nondeterministic (Hadamard) gates and $y \in \{0,1\}^h$.

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1, \ldots, x_n, y_1, \ldots, y_h, z_1, \ldots, z_n, w_1, \ldots, w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R'} \sum_{j=0}^{K-1} \omega^j (\#y, w : Q_C(x, y, z, w) = j)$$

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1,\dots,x_n,y_1,\dots,y_h,z_1,\dots,z_n,w_1,\dots,w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x,z \in \{0,1\}^n$:

$$\langle z \mid C \mid x\rangle = \frac{1}{R'}\sum_{j=0}^{K-1}\omega^j(\#y,w : Q_C(x,y,z,w)=j) = \frac{1}{R'}\sum_{y,w}\omega^{Q_C(x,y,z,w)},$$

where Q_C has the form $\sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$.

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1, \dots, x_n, y_1, \dots, y_h, z_1, \dots, z_n, w_1, \dots, w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R'} \sum_{j=0}^{K-1} \omega^j (\#y, w : Q_C(x, y, z, w) = j) = \frac{1}{R'} \sum_{y,w} \omega^{Q_C(x,y,z,w)},$$

where Q_C has the form $\sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$.

- Gives a particularly efficient reduction from BQP to #P.

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1, \dots, x_n, y_1, \dots, y_h, z_1, \dots, z_n, w_1, \dots, w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R'} \sum_{j=0}^{K-1} \omega^j (\#y, w : Q_C(x,y,z,w) = j) = \frac{1}{R'} \sum_{y,w} \omega^{Q_C(x,y,z,w)},$$

where Q_C has the form $\sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$.

- Gives a particularly efficient reduction from BQP to #P.
- In P_C, illegal paths that violate some constraint incur the value 0.

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1, \dots, x_n, y_1, \dots, y_h, z_1, \dots, z_n, w_1, \dots, w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R'} \sum_{j=0}^{K-1} \omega^j (\#y, w : Q_C(x,y,z,w) = j) = \frac{1}{R'} \sum_{y,w} \omega^{Q_C(x,y,z,w)},$$

where Q_C has the form $\sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$.

- Gives a particularly efficient reduction from BQP to #P.
- In P_C, illegal paths that violate some constraint incur the value 0.
- In Q_C, any violation creates an additive term $T = w_1 \cdots w_{\log_2 K}$ using fresh variables whose assignments give all values in $0 \mathrel{..} K-1$, which *cancel*.

Additive Case (Cf. Bacon-van Dam-Russell [2008])

Theorem (RC (2007-09), RCG (2018))

Given C and K, we can efficiently compute a polynomial $Q_C(x_1, \dots, x_n, y_1, \dots, y_h, z_1, \dots, z_n, w_1, \dots, w_t)$ of degree $O(1)$ over $\mathbb{Z}_K$ and a constant R' such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R'} \sum_{j=0}^{K-1} \omega^j (\#y, w : Q_C(x,y,z,w) = j) = \frac{1}{R'} \sum_{y,w} \omega^{Q_C(x,y,z,w)},$$

where Q_C has the form $\sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$.

- Gives a particularly efficient reduction from BQP to $\#\mathsf{P}$.
- In P_C, illegal paths that violate some constraint incur the value 0.
- In Q_C, any violation creates an additive term $T = w_1 \cdots w_{\log_2 K}$ using fresh variables whose assignments give all values in $0 \,..\, K-1$, which *cancel*. (This trick is my main original contribution.)

Constructing the Polynomials

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C \quad &*= \quad (1 - u_i y_j) \\ Q_C \quad &+= \quad 2^{k-1} u_i y_j. \end{aligned}$$

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\;*= & (1 - u_i y_j) \\ Q_C &\;+= & 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\mathrel{*}= (1 - u_i y_j) \\ Q_C &\mathrel{+}= 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$. No change to P_C or Q_C.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C \quad &*= \quad (1 - u_i y_j) \\ Q_C \quad &+= \quad 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$. No change to P_C or Q_C.
- S-gate: Q_C adds u_i^2.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\mathrel{*}= (1 - u_i y_j) \\ Q_C &\mathrel{+}= 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$. No change to P_C or Q_C.
- S-gate: Q_C adds u_i^2.
- CS-gate: Q_C adds $u_i u_j$.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\ *= & (1 - u_i y_j) \\ Q_C &\ += & 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_iu_j - u_i - u_j$. No change to P_C or Q_C.
- S-gate: Q_C adds u_i^2.
- CS-gate: Q_C adds u_iu_j.
- Thereby CS escapes the easy case over $\mathbb{Z}_4$ (with $k = 2$).

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\mathrel{*=} (1 - u_i y_j) \\ Q_C &\mathrel{+=} 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_iu_j - u_i - u_j$. No change to P_C or Q_C.
- S-gate: Q_C adds u_i^2.
- CS-gate: Q_C adds u_iu_j.
- Thereby CS escapes the easy case over $\mathbb{Z}_4$ (with $k = 2$).
- TOF: controls u_i, u_j stay, target u_k changes to $2u_iu_ju_k - u_iu_j - u_k$.

Constructing the Polynomials

- Initially $P_C = 1$, $Q_C = 0$.
- For Hadamard on line i (u_i—H–), allocate new variable y_j and do:

$$\begin{aligned} P_C &\;*=\; (1 - u_i y_j) \\ Q_C &\;+=\; 2^{k-1} u_i y_j. \end{aligned}$$

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_iu_j - u_i - u_j$. No change to P_C or Q_C.
- S-gate: Q_C adds u_i^2.
- CS-gate: Q_C adds u_iu_j.
- Thereby CS escapes the easy case over $\mathbb{Z}_4$ (with $k = 2$).
- TOF: controls u_i, u_j stay, target u_k changes to $2u_iu_ju_k - u_iu_j - u_k$.
- T-gate also goes cubic.

Logical Simulation

Theorem (C. Guan in RCG 2018)

Given C, n, K, h as above, we can quickly build a Boolean formula ϕ_C in variables $y_1, \ldots, y_h$, together with substituted-for $x_1, \ldots, x_n$, $z_1, \ldots, z_n$, and other "forced" variables such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j \cdot \#sat(\phi_C).$$

Logical Simulation

Theorem (C. Guan in RCG 2018)

Given C, n, K, h as above, we can quickly build a Boolean formula ϕ_C in variables $y_1, \ldots, y_h$, together with substituted-for $x_1, \ldots, x_n$, $z_1, \ldots, z_n$, and other "forced" variables such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j \cdot \#sat(\phi_C).$$

- The ϕ is a conjunction of "controlled bitflips" $p' = p \oplus (u \wedge v)$.

Logical Simulation

Theorem (C. Guan in RCG 2018)

Given C, n, K, h as above, we can quickly build a Boolean formula ϕ_C in variables $y_1, \ldots, y_h$, together with substituted-for $x_1, \ldots, x_n$, $z_1, \ldots, z_n$, and other "forced" variables such that for all $x, z \in \{0,1\}^n$:

$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j \cdot \#sat(\phi_C).$$

- The ϕ is a conjunction of "controlled bitflips" $p' = p \oplus (u \wedge v)$.
- Easy to transform into 3CNF (i.e., "3SAT" form). (show demo)

Logical Simulation

Theorem (C. Guan in RCG 2018)

Given C, n, K, h as above, we can quickly build a Boolean formula ϕ_C in variables $y_1, \ldots, y_h$, together with substituted-for $x_1, \ldots, x_n$, $z_1, \ldots, z_n$, and other "forced" variables such that for all $x, z \in \{0,1\}^n$:

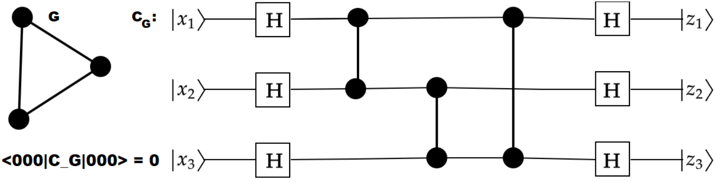
$$\langle z \mid C \mid x \rangle = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j \cdot \#sat(\phi_C).$$

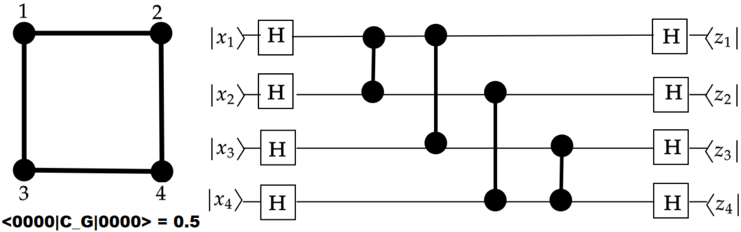
- The ϕ is a conjunction of "controlled bitflips" $p' = p \oplus (u \wedge v)$.
- Easy to transform into 3CNF (i.e., "3SAT" form). **(show demo)**
- For $K = 2, 4$ (i.e., for $\mathsf{H} + \mathsf{Tof}$ and $\mathsf{H} + \mathsf{CS}$), we get the acceptance *probability* as a simple difference:

$$|\langle z \mid C \mid x \rangle|^2 = \frac{1}{R} \left(\#sat(\phi_C) - \#sat(\phi'_C) \right).$$

II. Strong Simulation of Graph State Circuits

Computing amplitudes $\langle z \mid C \mid x \rangle$ for Clifford circuits C can be efficiently reduced to computing $\langle 0^n \mid C_G \mid 0^n \rangle$ for **graph-state circuits** C_G of graphs G, using H and CZ gates, as exemplified by:





Improved From $O(n^3)$ to $O(n^{2.37155...})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x\rangle$ can be computed in $O(s + n^{\omega})$ time, where $\omega \leq 2.37155...$ is the exponent of multiplying $n \times n$ matrices.

Improved From $O(n^3)$ to $O(n^{2.37155...})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x \rangle$ *can be computed in* $O(s + n^{\omega})$ *time, where* $\omega \leq 2.37155...$ *is the exponent of multiplying* $n \times n$ *matrices.*

- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].

Improved From $O(n^3)$ to $O(n^{2.37155...})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x \rangle$ can be computed in $O(s + n^{\omega})$ time, where $\omega \leq 2.37155...$ is the exponent of multiplying $n \times n$ matrices.

- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].
- **Corollary:** Counting solutions to quadratic polynomials $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_2$ is in $O(n^{2.37155...})$ time.

Improved From $O(n^3)$ to $O(n^{2.37155\dots})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x \rangle$ can be computed in $O(s + n^\omega)$ time, where $\omega \leq 2.37155\dots$ is the exponent of multiplying $n \times n$ matrices.

- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].
- **Corollary:** Counting solutions to quadratic polynomials $p(x_1, \dots, x_n)$ over $\mathbb{Z}_2$ is in $O(n^{2.37155\dots})$ time.

Improved From $O(n^3)$ to $O(n^{2.37155\ldots})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x \rangle$ can be computed in $O(s + n^\omega)$ time, where $\omega \leq 2.37155\ldots$ is the exponent of multiplying $n \times n$ matrices.

- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].
- **Corollary:** Counting solutions to quadratic polynomials $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_2$ is in $O(n^{2.37155\ldots})$ time.
- Improves $O(n^3)$ time of Ehrenfeucht-Karpinski (1990).

Improved From $O(n^3)$ to $O(n^{2.37155...})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x \rangle$ can be computed in $O(s + n^\omega)$ time, where $\omega \leq 2.37155...$ is the exponent of multiplying $n \times n$ matrices.

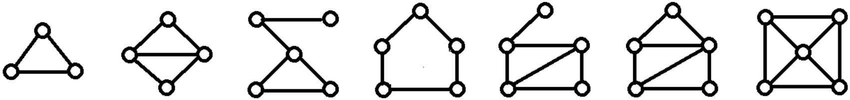
- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].
- **Corollary:** Counting solutions to quadratic polynomials $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_2$ is in $O(n^{2.37155...})$ time.
- Improves $O(n^3)$ time of Ehrenfeucht-Karpinski (1990).
- See Beaudrap and Herbert [2021] for other time/size/#H tradeoffs.

Improved From $O(n^3)$ to $O(n^{2.37155...})$

Theorem (Guan-Regan, 2019)

For n-qubit stabilizer circuits of size s, $\langle z \mid C \mid x\rangle$ can be computed in $O(s + n^{\omega})$ time, where $\omega \leq 2.37155...$ is the exponent of multiplying $n \times n$ matrices.

- Although C has $K = 2$, proof needs to use quadratic forms over $\mathbb{Z}_4$. And LDU decompositions over $\mathbb{Z}_2$ by Dumas-Pernet [2018].
- **Corollary:** Counting solutions to quadratic polynomials $p(x_1, \dots, x_n)$ over $\mathbb{Z}_2$ is in $O(n^{2.37155...})$ time.
- Improves $O(n^3)$ time of Ehrenfeucht-Karpinski (1990).
- See Beaudrap and Herbert [2021] for other time/size/#H tradeoffs.
- Can we recognize G with $\langle 0^n \mid C_G \mid 0^n\rangle = 0$ more quickly still?



From Graphs to Polymatroids

From Graphs to Polymatroids

- A self-loop on node i becomes a Z-gate on qubit line i.

From Graphs to Polymatroids

- A self-loop on node i becomes a Z-gate on qubit line i.
- An S-gate on line i would then be a "half loop."

From Graphs to Polymatroids

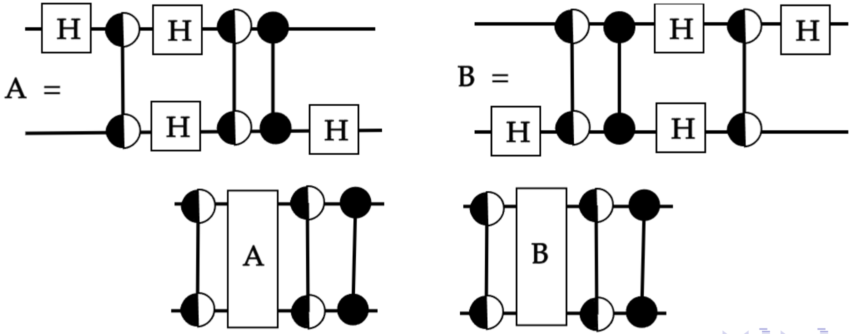
- A self-loop on node i becomes a Z-gate on qubit line i.
- An S-gate on line i would then be a "half loop."
- A CS gate would then be a "half edge."

From Graphs to Polymatroids

- A self-loop on node i becomes a Z-gate on qubit line i.
- An S-gate on line i would then be a "half loop."
- A CS gate would then be a "half edge."
- Formalizable as a **polymatroid** (PM). Into universal QC now.

From Graphs to Polymatroids

- A self-loop on node i becomes a Z-gate on qubit line i.
- An S-gate on line i would then be a "half loop."
- A CS gate would then be a "half edge."
- Formalizable as a **polymatroid** (PM). Into universal QC now.
- John Preskill's notes show that the following four widgets, together with their conjugations by $\mathsf{H} \otimes \mathsf{H}$, suffice:



New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.

New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?

New Heuristic Forms to Investigate

- Would be a “PM State Circuit”—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?

New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?
- How about the power of PM state circuits by themselves?

New Heuristic Forms to Investigate

- Would be a “PM State Circuit”—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?
- How about the power of PM state circuits by themselves?
- Are they more amenable to algebraic or logical model-counting heuristics than general quantum circuits?

New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?
- How about the power of PM state circuits by themselves?
- Are they more amenable to algebraic or logical model-counting heuristics than general quantum circuits?
- Chaowen and I also considered graphs that can have:
    - Loops not attached to a vertex, called *circles*.
    - Numbered copies of the empty graph, called *wisps*.
    - Wisps of negative sign, called *negative isols*.

New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?
- How about the power of PM state circuits by themselves?
- Are they more amenable to algebraic or logical model-counting heuristics than general quantum circuits?
- Chaowen and I also considered graphs that can have:
  - Loops not attached to a vertex, called *circles*.
  - Numbered copies of the empty graph, called *wisps*.
  - Wisps of negative sign, called *negative isols*.
- They can be formalized via (*graphical*) 2-polymatroids. Call them "(G)2PMs."

New Heuristic Forms to Investigate

- Would be a "PM State Circuit"—except for all those H gates in the middle.
- Can we move them to the sides, as with graph state circuits?
- If not, are there other useful canonical forms, a-la this?
- How about the power of PM state circuits by themselves?
- Are they more amenable to algebraic or logical model-counting heuristics than general quantum circuits?
- Chaowen and I also considered graphs that can have:
  - Loops not attached to a vertex, called *circles*.
  - Numbered copies of the empty graph, called *wisps*.
  - Wisps of negative sign, called *negative isols*.
- They can be formalized via (*graphical*) 2-polymatroids. Call them "(G)2PMs."
- We took them in a different direction.

Singular Value Decomposition

Unlike with diagonalization, this is *always* possible:

SVD Theorem: For every $m \times n$ matrix A we can efficiently find:

- an $m \times m$ unitary matrix U,
- an $m \times n$ pseudo-diagonal matrix Σ with non-negative entries $\Sigma[i,i] = \sigma_i$, and
- an $n \times n$ unitary matrix V,

such that $A = U\Sigma V^*$. Furthermore, we can arrange that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)}$, and in consequence:

- $||A||_F = \sqrt{\sum_i \sigma_i^2}$,
- $||A||_2 = \sigma_1$,
- $A^*A = V\Sigma^T U^* U \Sigma V^* = V \operatorname{diag}\left(\sigma_i^2\right) V^*$, and
- $AA^* = U\Sigma V^* V \Sigma^T U^* = U \operatorname{diag}\left(\sigma_i^2\right) U^*$,

so that the squares of the σ_i and associated vectors give the spectral decompositions of the Hermitian PSD matrices A^*A and AA^*, respectively.

III. SVD and Tensor Network Ideas

III. SVD and Tensor Network Ideas

- Show SVD and truncation idea.

III. SVD and Tensor Network Ideas

- Show SVD and truncation idea.
- SVD in Image Compression.

III. SVD and Tensor Network Ideas

- Show SVD and truncation idea.
- SVD in Image Compression.
- Strategy for simulating quantum circuits via *classical* **tensor networks** and **SVD truncation**.

III. SVD and Tensor Network Ideas

- Show SVD and truncation idea.
- SVD in Image Compression.
- Strategy for simulating quantum circuits via *classical* **tensor networks** and **SVD truncation**.
- Most workable scheme?

III. SVD and Tensor Network Ideas

- Show SVD and truncation idea.
- SVD in Image Compression.
- Strategy for simulating quantum circuits via *classical* **tensor networks** and **SVD truncation**.
- Most workable scheme?
- How this might be programmed.

What Is the Status?

What Is the Status?

- Can quantum hardware open up and widen a gap over classical?

What Is the Status?

- Can quantum hardware open up and widen a gap over classical?
- Can more-clever classical simulations always catch up?

What Is the Status?

- Can quantum hardware open up and widen a gap over classical?
- Can more-clever classical simulations always catch up?
- What does Nature do, anyway? Is it the "rose" in Umberto Eco's maxim *Stat rosa pristina nomine, nomina nuda tenemus*—?

What Is the Status?

- Can quantum hardware open up and widen a gap over classical?
- Can more-clever classical simulations always catch up?
- What does Nature do, anyway? Is it the "rose" in Umberto Eco's maxim *Stat rosa pristina nomine, nomina nuda tenemus*—?
- If so, are human brains left behind with Turing's vision? In verse:

What Is the Status?

- Can quantum hardware open up and widen a gap over classical?
- Can more-clever classical simulations always catch up?
- What does Nature do, anyway? Is it the "rose" in Umberto Eco's maxim *Stat rosa pristina nomine, nomina nuda tenemus*—?
- If so, are human brains left behind with Turing's vision? In verse:

"It From Bit" we once proclaimed,
but now the Bit has bit the dust
of whizzing quantum chips that gamed
coherence, to evade the trust
that the Word framed creation's hour:
Mother Nature fully lexical.
Why not evolve us that same power?
It is a status most perplexical.