Tracking Quantum Circuits By Polynomials

Oxford University OASIS Seminar

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

12 June, 2015

[1] Includes joint work with Amlan Chakrabarti, U. Calcutta

Boolean Circuits Have. . .

Boolean Circuits Have...

- n inputs $x_1, \ldots, x_n \in \{0, 1\}^n$

Boolean Circuits Have. . .

- n inputs $x_1, \ldots, x_n \in \{0, 1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$

Boolean Circuits Have. . .

- n inputs $x_1, \ldots, x_n \in \{0, 1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$
- Maybe h-many nondeterministic inputs $y_1, \ldots, y_h$?

Boolean Circuits Have...

- n inputs $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$
- Maybe h-many nondeterministic inputs $y_1, \ldots, y_h$?
- m gates $g_1, \ldots, g_m$ (wlog. all NAND)

Boolean Circuits Have...

- n inputs $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$
- Maybe h-many nondeterministic inputs $y_1, \ldots, y_h$?
- m gates $g_1, \ldots, g_m$ (wlog. all NAND)
- Up to $2m + r$ wires (if fan-in ≤ 2)

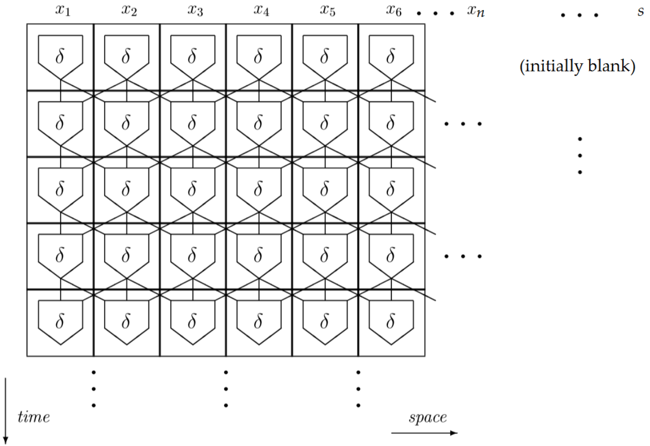
Boolean Circuits Have...

- n inputs $x_1, \ldots, x_n \in \{0, 1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$
- Maybe h-many nondeterministic inputs $y_1, \ldots, y_h$?
- m gates $g_1, \ldots, g_m$ (wlog. all NAND)
- Up to $2m + r$ wires (if fan-in ≤ 2)
- Each wire has a definite 0-1 value.

Boolean Circuits Have...

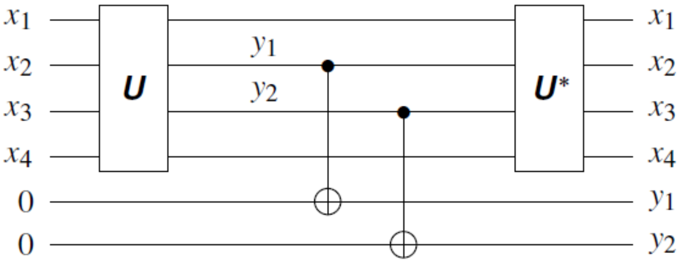
- n inputs $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ outputs $z_1, \ldots, z_r$
- Maybe h-many nondeterministic inputs $y_1, \ldots, y_h$?
- m gates $g_1, \ldots, g_m$ (wlog. all NAND)
- Up to $2m + r$ wires (if fan-in ≤ 2)
- Each wire has a definite 0-1 value.
- Bits have no common identity across wires, but they *can*...

Turing "Cue Bits"



Space s, so $n - s$ "ancillary" cells.

Quantum Circuits: similar picture



Example also shows the **copy-uncompute trick**.

Quantum Circuits Have. . .

Quantum Circuits Have. . .

- n input **qubits** $x_1, \ldots, x_n \in \{0, 1\}^n$

Quantum Circuits Have...

- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)

Quantum Circuits Have...

- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits

Quantum Circuits Have...

- n input **qubits** $x_1, \ldots, x_n \in \{0, 1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits
- m-many **quantum gates** (arities can be 1,2,3)

Quantum Circuits Have. . .

- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits
- m-many **quantum gates** (arities can be 1,2,3)
- Maybe h of them are **Hadamard gates**, which supply nondeterminism.

Quantum Circuits Have...

- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits
- m-many **quantum gates** (arities can be 1,2,3)
- Maybe h of them are **Hadamard gates**, which supply nondeterminism.
- Qubits retain identity as wires transit gates.

Quantum Circuits Have...

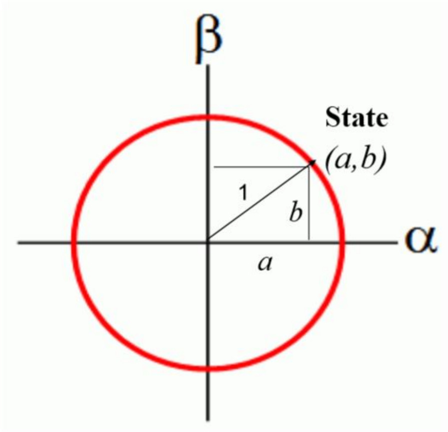
- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits
- m-many **quantum gates** (arities can be 1,2,3)
- Maybe h of them are **Hadamard gates**, which supply nondeterminism.
- Qubits retain identity as wires transit gates.
- Each wire need **not** have a definite 0-1 value, owing to **entanglement**.

Quantum Circuits Have...

- n input **qubits** $x_1, \ldots, x_n \in \{0,1\}^n$
- $r \geq 1$ output qubits $z_1, \ldots, z_r$ (think $r = 1$ or $r = n$)
- $s - n$ **ancilla** qubits
- m-many **quantum gates** (arities can be 1,2,3)
- Maybe h of them are **Hadamard gates**, which supply nondeterminism.
- Qubits retain identity as wires transit gates.
- Each wire need **not** have a definite 0-1 value, owing to **entanglement**.
- Under the hood are(??) $S = 2^s$ complex entries of a unit **state vector**.

A Qubit

Quantum Bits, e.g. spins.



Probability of observing Alpha is a-squared, Beta is b-squared. By Pythagoras, these add to 1.

Quantum Gates

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:
- $\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ identity

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:
- $\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ identity
- $\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ negation, aka. NOT

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:
- $\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ identity
- $\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ negation, aka. NOT
- $\mathsf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -11 \end{bmatrix}$ Hadamard gate.

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:
- $\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ identity
- $\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ negation, aka. NOT
- $\mathsf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -11 \end{bmatrix}$ Hadamard gate.
- Only non-permutation gate needed for universality.

Quantum Gates

- A k-ary gate can be represented by a $K \times K$ **unitary** matrix, $K = 2^k$.
- Common gates for $k = 1$, $K = 2$:
- $\mathsf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ identity
- $\mathsf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ negation, aka. NOT
- $\mathsf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -11 \end{bmatrix}$ Hadamard gate.
- Only non-permutation gate needed for universality.
- But also common: $\mathsf{Y} = \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix}$, $\mathsf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$, $\mathsf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$.

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation (1 2 4 3),

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation $(1\ 2\ 4\ 3)$, swap $(3\ 4)$.

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation (1 2 4 3), swap (3 4). Also called CX.

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation (1 2 4 3), swap (3 4). Also called CX.

$$\mathsf{CNOT} \circ (\mathsf{H} \otimes \mathsf{I}) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix}$$

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation (1 2 4 3), swap (3 4). Also called CX.

$$\mathsf{CNOT} \circ (\mathsf{H} \otimes \mathsf{I}) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix}$$

Applied to $e_{00} = (1, 0, 0, 0)^T$ gives $\frac{1}{\sqrt{2}}(e_{00} + e_{11})$.

Binary Gates

With $k = 2$ qubits, $K = 4$. "Controlled Not" showing quantum coordinates:

$$\mathsf{CNOT} = \begin{array}{c|cccc|} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Permutation $(1\ 2\ 4\ 3)$, swap $(3\ 4)$. Also called CX.

$$\mathsf{CNOT} \circ (\mathsf{H} \otimes \mathsf{I}) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix}$$

Applied to $e_{00} = (1,0,0,0)^T$ gives $\frac{1}{\sqrt{2}}(e_{00} + e_{11})$. **EPR Entanglement.**

Ternary Toffoli Gate: $K = 8$

Ternary Toffoli Gate: $K = 8$

- TOF $= diag(1, 1, 1, 1, 1, 1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Ternary Toffoli Gate: $K = 8$

- $\mathsf{TOF} = diag(1, 1, 1, 1, 1, 1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.
- Fixes $000, \ldots, 101$; swaps $110 \leftrightarrow 111$.

Ternary Toffoli Gate: $K = 8$

- $\mathsf{TOF} = diag(1,1,1,1,1,1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.
- Fixes $000, \ldots, 101$; swaps $110 \leftrightarrow 111$.
- Control-Control-NOT, hence also called CCX.

Ternary Toffoli Gate: $K = 8$

- $\mathsf{TOF} = diag(1,1,1,1,1,1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.
- Fixes $000, \ldots, 101$; swaps $110 \leftrightarrow 111$.
- Control-Control-NOT, hence also called CCX.
- $\mathsf{TOF}(a, b, 1) = (-, -, a \text{ NAND } b)$, Thus TOF is classically universal.

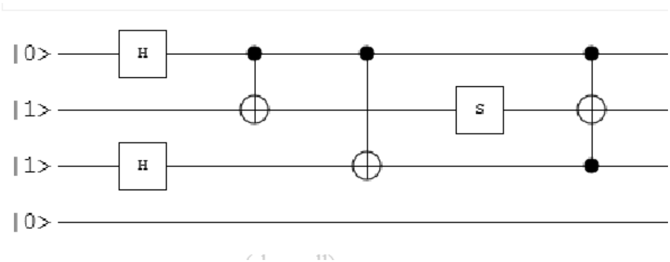
Ternary Toffoli Gate: $K = 8$

- $\mathsf{TOF} = diag(1,1,1,1,1,1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.
- Fixes $000, \ldots, 101$; swaps $110 \leftrightarrow 111$.
- Control-Control-NOT, hence also called CCX.
- $\mathsf{TOF}(a,b,1) = (-,-,a \text{ NAND } b)$, Thus TOF is classically universal.
- $\mathsf{H} + \mathsf{TOF}$ is quantum universal.

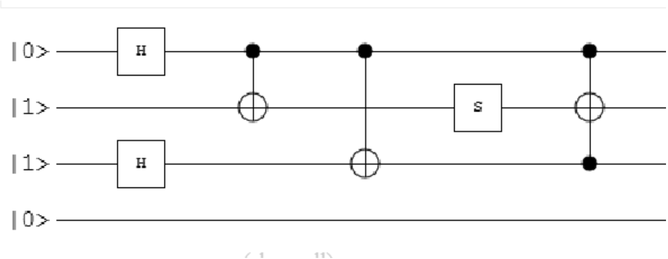
Ternary Toffoli Gate: $K = 8$

- $\mathsf{TOF} = diag(1,1,1,1,1,1)$, then $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.
- Fixes $000, \ldots, 101$; swaps $110 \leftrightarrow 111$.
- Control-Control-NOT, hence also called CCX.
- $\mathsf{TOF}(a,b,1) = (-,-,a \text{ NAND } b)$, Thus TOF is classically universal.
- $\mathsf{H} + \mathsf{TOF}$ is quantum universal.
- $\mathsf{H} + \mathsf{CNOT}$ is not quantum universal; it recognizes a proper subclass of P.

Example Quantum Circuit

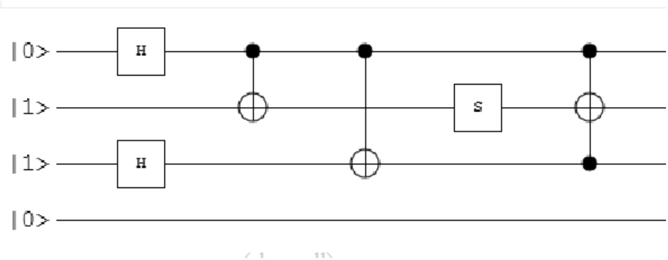


Example Quantum Circuit



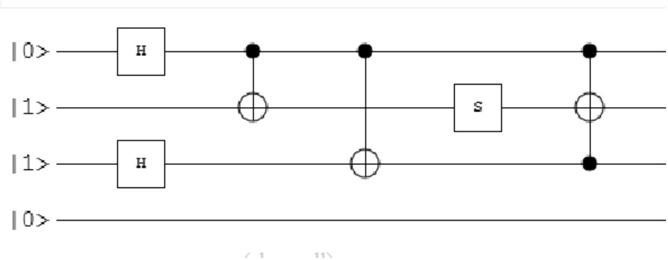
1. $\mathsf{H} \otimes \mathsf{I} \otimes \mathsf{H} \otimes \mathsf{I}^{\otimes(s-3)}$.

Example Quantum Circuit



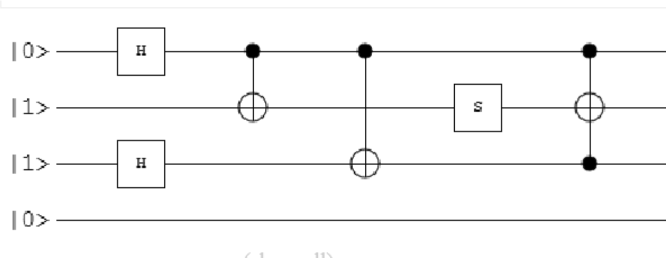
1. $\mathsf{H} \otimes \mathsf{I} \otimes \mathsf{H} \otimes \mathsf{I}^{\otimes(s-3)}$.
2. $\mathsf{CNOT} \otimes \mathsf{I}^{\otimes(s-2)}$. First three lines have "CXI."

Example Quantum Circuit



1. $\mathsf{H} \otimes \mathsf{I} \otimes \mathsf{H} \otimes \mathsf{I}^{\otimes(s-3)}$.
2. $\mathsf{CNOT} \otimes \mathsf{I}^{\otimes(s-2)}$. First three lines have "CXI."
3. "CIX"—semantically but not syntactically $\otimes$ of I and CNOT.

Example Quantum Circuit



1. $\mathsf{H} \otimes \mathsf{I} \otimes \mathsf{H} \otimes \mathsf{I}^{\otimes(s-3)}$.
2. $\mathsf{CNOT} \otimes \mathsf{I}^{\otimes(s-2)}$. First three lines have "CXI."
3. "CIX"—semantically but not syntactically $\otimes$ of I and CNOT.
4. After the S in stage 4, a TOF with controls on 1,3 and target on 2. The whole C computes a unitary U_C.

Input-Output and Measurement

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}}$

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \ldots, z_{S-1})$, $S = 2^s$.

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \ldots, z_{S-1})$, $S = 2^s$.
- Measure all lines: For any outcome $b \in \{0,1\}^s$, $\Pr[C(x) \to b] = |z_b|^2 = |\langle E_x U_C e_b \rangle|^2$.

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \ldots, z_{S-1})$, $S = 2^s$.
- Measure all lines: For any outcome $b \in \{0,1\}^s$, $\Pr[C(x) \to b] = |z_b|^2 = |\langle E_x U_C e_b \rangle|^2$.
- (Show how DavyW applet does this.)

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \ldots, z_{S-1})$, $S = 2^s$.
- Measure all lines: For any outcome $b \in \{0,1\}^s$,
  $\Pr[C(x) \to b] = |z_b|^2 = |\langle E_x U_C e_b \rangle|^2$.
- (Show how DavyW applet does this.)
- For outcome $d \in \{0,1\}^r$ on r-many designated qubit lines,
  $\Pr[C(x) \to d] = \sum_{b \sqsupseteq d} |z_b|^2$.

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \dots, z_{S-1})$, $S = 2^s$.
- Measure all lines: For any outcome $b \in \{0,1\}^s$,
  $\Pr[C(x) \to b] = |z_b|^2 = |\langle E_x U_C e_b \rangle|^2$.
- (Show how DavyW applet does this.)
- For outcome $d \in \{0,1\}^r$ on r-many designated qubit lines,
  $\Pr[C(x) \to d] = \sum_{b \sqsupseteq d} |z_b|^2$.
- Can project as amplitudes: $C(x) \mapsto (z'_0, \dots, z'_{2^r - 1})$ where $|z_d|^2$ is the probability of outcome $d \in \{0,1\}^r$.

Input-Output and Measurement

- Input: $E_x = e_{x0^{n-s}} = e_{x_1} \otimes e_{x_2} \otimes \cdots \otimes e_{x_n} \otimes e_0^{\otimes(n-s)}$.
- Output: A state vector $(z_0, \dots, z_{S-1})$, $S = 2^s$.
- Measure all lines: For any outcome $b \in \{0,1\}^s$, $\Pr[C(x) \to b] = |z_b|^2 = |\langle E_x U_C e_b \rangle|^2$.
- (Show how DavyW applet does this.)
- For outcome $d \in \{0,1\}^r$ on r-many designated qubit lines, $\Pr[C(x) \to d] = \sum_{b \sqsupseteq d} |z_b|^2$.
- Can project as amplitudes: $C(x) \mapsto (z'_0, \dots, z'_{2^r-1})$ where $|z_d|^2$ is the probability of outcome $d \in \{0,1\}^r$.
- Call this amplitude z_d as $A[C(x) \mapsto d]$.

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$\begin{aligned} x \in L &\implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4}, \\ x \notin L &\implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4}, \end{aligned}$$

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$\begin{aligned} x \in L &\implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4}, \\ x \notin L &\implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4}, \end{aligned}$$

- FACT: FACT $\in$ BQP.

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$x \in L \implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4},$$
$$x \notin L \implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4},$$

- FACT: FACT $\in$ BQP. (Say FACT $= \{(x, w) : w \sqsubseteq UPF(x)\}$.)

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$\begin{aligned} x \in L &\implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4}, \\ x \notin L &\implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4}, \end{aligned}$$

- FACT: FACT $\in$ BQP. (Say FACT $= \{(x,w) : w \sqsubseteq UPF(x)\}$.)
- P $\subseteq$ BPP $\subseteq$ BQP $\subseteq$ PP $\equiv_T^p$ #P.

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$\begin{aligned} x \in L &\implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4}, \\ x \notin L &\implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4}, \end{aligned}$$

- FACT: FACT $\in$ BQP. (Say FACT $= \{(x, w) : w \sqsubseteq UPF(x)\}$.)
- P $\subseteq$ BPP $\subseteq$ BQP $\subseteq$ PP $\equiv_T^p$ #P.
- No evidence for NP $\subseteq$ BQP,

BQP

Definition

A language L belongs to BQP if there are poly-time uniform quantum circuits C_n for each n such that forall n and inputs $x \in \{0,1\}^n$, designating qubit 1 for yes/no output:

$$\begin{aligned} x \in L &\implies \Pr[C_n(x) \mapsto 1] > \frac{3}{4}, \\ x \notin L &\implies \Pr[C_n(x) \mapsto 1] < \frac{1}{4}, \end{aligned}$$

- FACT: FACT $\in$ BQP. (Say FACT $= \{(x, w) : w \sqsubseteq UPF(x)\}$.)
- $\mathsf{P} \subseteq \mathsf{BPP} \subseteq \mathsf{BQP} \subseteq \mathsf{PP} \equiv_T^p \#\mathsf{P}$.
- No evidence for $\mathsf{NP} \subseteq \mathsf{BQP}$, nor $\mathsf{BQP} \subseteq \mathsf{PH}$.

What to Represent By Polynomials?

1. The acceptance probability $p_x = \Pr[C(x) \mapsto 1]$?

What to Represent By Polynomials?

1. The acceptance probability $p_x = \Pr[C(x) \mapsto 1]$? So we convert C into a polynomial p_C such that for all x,

$$p_x = p_C(x_1, \ldots, x_n).$$

Used in quantum query complexity lower bounds, but *building* p_C is hard.

What to Represent By Polynomials?

1. The acceptance probability $p_x = \Pr[C(x) \mapsto 1]$? So we convert C into a polynomial p_C such that for all x,

$$p_x = p_C(x_1, \ldots, x_n).$$

Used in quantum query complexity lower bounds, but *building* p_C is hard.

2. The acceptance amplitude, but *implicitly* by counting zeroes. Given a polynomial $P(x_1, \ldots, x_n, y_1, \ldots, y_h)$, define for all $x \in \{0,1\}^n$ and value a:

What to Represent By Polynomials?

1. The acceptance probability $p_x = \Pr[C(x) \mapsto 1]$? So we convert C into a polynomial p_C such that for all x,

$$p_x = p_C(x_1, \ldots, x_n).$$

Used in quantum query complexity lower bounds, but *building* p_C is hard.

2. The acceptance amplitude, but *implicitly* by counting zeroes. Given a polynomial $P(x_1, \ldots, x_n, y_1, \ldots, y_h)$, define for all $x \in \{0,1\}^n$ and value a:

$$N_{P,x}[a] = |\{y \in \{0,1\}^h : P(x,y) = a\}|.$$

This is a $\#\mathsf{P}$ function.

The Basic Theorem

Theorem (Dawson et al.. 2004, implicitly before?)

Given C *built from* TOF *gates and* h*-many* H *gates, we can efficiently compute a polynomial* $P_C(x_1, \dots, x_n, y_1, \dots, y_h, z_1, \dots, z_r)$ *and a constant* R *(here,* $R = \sqrt{2^h}$*) such that for all* $x \in \{0,1\}^n$ *and* $z \in \{0,1\}^r$*,*

$$A[C(x) \mapsto z] = \frac{1}{R}[N_{P,x,z}[1] - N_{P,x,z}[0].$$

The Basic Theorem

Theorem (Dawson et al.. 2004, implicitly before?)

Given C built from TOF *gates and h-many* H *gates, we can efficiently compute a polynomial $P_C(x_1, \ldots, x_n, y_1, \ldots, y_h, z_1, \ldots, z_r)$ and a constant R (here, $R = \sqrt{2^h}$) such that for all $x \in \{0,1\}^n$ and $z \in \{0,1\}^r$,*

$$A[C(x) \mapsto z] = \frac{1}{R}[N_{P,x,z}[1] - N_{P,x,z}[0].$$

- Thus BQP reduces to the difference between to #P functions.

The Basic Theorem

Theorem (Dawson et al.. 2004, implicitly before?)

Given C built from TOF *gates and h-many* H *gates, we can efficiently compute a polynomial $P_C(x_1, \ldots, x_n, y_1, \ldots, y_h, z_1, \ldots, z_r)$ and a constant R (here, $R = \sqrt{2^h}$) such that for all $x \in \{0,1\}^n$ and $z \in \{0,1\}^r$,*

$$A[C(x) \mapsto z] = \frac{1}{R}[N_{P,x,z}[1] - N_{P,x,z}[0].$$

- Thus BQP reduces to the difference between to #P functions.
- Note heavy promise: $0 \leq N[1] - N[0] \leq R = \sqrt{2^h}$.

The Basic Theorem

Theorem (Dawson et al.. 2004, implicitly before?)

Given C built from TOF *gates and h-many* H *gates, we can efficiently compute a polynomial $P_C(x_1,\dots,x_n,y_1,\dots,y_h,z_1,\dots,z_r)$ and a constant R (here, $R=\sqrt{2^h}$) such that for all $x\in\{0,1\}^n$ and $z\in\{0,1\}^r$,*

$$A[C(x)\mapsto z]=\frac{1}{R}[N_{P,x,z}[1]-N_{P,x,z}[0].$$

- Thus BQP reduces to the difference between to #P functions.
- Note heavy promise: $0\le N[1]-N[0]\le R=\sqrt{2^h}$.
- Means all but a trace of pairs $y,y'\in\{0,1\}^h$ cancel.

The Basic Theorem

Theorem (Dawson et al.. 2004, implicitly before?)

Given C built from TOF *gates and h-many* H *gates, we can efficiently compute a polynomial $P_C(x_1,\dots,x_n,y_1,\dots,y_h,z_1,\dots,z_r)$ and a constant R (here, $R=\sqrt{2^h}$) such that for all $x\in\{0,1\}^n$ and $z\in\{0,1\}^r$,*

$$A[C(x)\mapsto z]=\frac{1}{R}[N_{P,x,z}[1]-N_{P,x,z}[0].$$

- Thus BQP reduces to the difference between to #P functions.
- Note heavy promise: $0\leq N[1]-N[0]\leq R=\sqrt{2^h}$.
- Means all but a trace of pairs $y,y'\in\{0,1\}^h$ cancel.
- Hence cannot simply use Stockmeyer's approximation of counting to get $\mathsf{BQP}\subseteq\Sigma_3^p\cap\Pi_3^p$.

My Extensions

My Extensions

- Say a gate is *balanced* if all nonzero entries $re^{i\theta}$ of its matrix have equal magnitude $|r|$.

My Extensions

- Say a gate is *balanced* if all nonzero entries $re^{i\theta}$ of its matrix have equal magnitude $|r|$.
- A circuit C is balanced if every gate in C is balanced.

My Extensions

- Say a gate is *balanced* if all nonzero entries $re^{i\theta}$ of its matrix have equal magnitude $|r|$.
- A circuit C is balanced if every gate in C is balanced.
- $K(C) =$ the least K such that all θ in entries of gates in C are multiples of $2\pi/K$. "Min-Phase"

My Extensions

- Say a gate is *balanced* if all nonzero entries $re^{i\theta}$ of its matrix have equal magnitude $|r|$.
- A circuit C is balanced if every gate in C is balanced.
- $K(C)$ = the least K such that all θ in entries of gates in C are multiples of $2\pi/K$. "Min-Phase"
- Let G be a field or ring such that G^* embeds the K-th roots of unity ω^j by a multiplicative homomorphism $e(\omega^j)$.

Theorem

Can arrange $P_C = \prod_{gates\ g} P_g$ *such that for all* x *and* z,

$$A[C(x) \mapsto z] = \frac{1}{R} \sum_{j=0}^{K-1} \omega^j N_{P_C,x,z}[e(\omega^j)]$$

My Extensions

- Say a gate is *balanced* if all nonzero entries $re^{i\theta}$ of its matrix have equal magnitude $|r|$.
- A circuit C is balanced if every gate in C is balanced.
- $K(C)$ = the least K such that all θ in entries of gates in C are multiples of $2\pi/K$. "Min-Phase"
- Let G be a field or ring such that G^* embeds the K-th roots of unity ω^j by a multiplicative homomorphism $e(\omega^j)$.

Theorem

Can arrange $P_C = \prod_{gates\ g} P_g$ *such that for all* x *and* z,

$$A[C(x) \mapsto z] = \frac{1}{R}\sum_{j=0}^{K-1} \omega^j N_{P_C,x,z}[e(\omega^j)] = \frac{1}{R}\sum_{y} \omega^{P_C(x,y,z)}.$$

Additive Extension

Theorem

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

$$A[C(x) \mapsto z] = \frac{1}{R}\sum_{j=0}^{K-1} \omega^j N_{Q_C,x,z}[j]$$

Additive Extension

Theorem

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

$$A[C(x) \mapsto z] = \frac{1}{R}\sum_{j=0}^{K-1} \omega^j N_{Q_C,x,z}[j] = \frac{1}{R}\sum_{y,w} \omega^{Q_C(x,y,z,w)},$$

where $Q_C = \sum_{gates\ g} q_g + \sum_{constraints\ c} q_c$ has bounded degree.

My trick: Given a **constraint** c with values $0 =$ fail, $1 =$ OK, add

$$q_c = w_0(1-c) + 2w_1(1-c) + 4w_2(1-c) + \cdots + 2^{k-1}w_{k-1}(1-c).$$

Then $c = 0 \implies$ binary assignments to $w_0,\dots,w_{k-1}$ run through all K values $\implies$ the entire sum over y,w cancels. Whereas $c = 1$ zeroes all such terms, so he only effect is to inflate R.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

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

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

- 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, as with any permutation gate.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$.
- No change to P_C or Q_C, as with any permutation gate.
- In characteristic 2, linearity is preserved.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

- CNOT with incoming terms u_i on control, u_j on target: u_i stays, $u_j := 2u_i u_j - u_i - u_j$.
- No change to P_C or Q_C, as with any permutation gate.
- In characteristic 2, linearity is preserved.
- TOF: controls u_i, u_j stay, target u_k changes to $2u_i u_j u_k - u_i u_j - u_k$.

Computing the Polynomials

- "Annotate" every juncture of qubit i with variable y_j or term u_i.
- Initially x_i and 0 terms, $P_C = 1$, $Q_C = 0$.
- u_i—H—: new variable y_j,

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

- 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, as with any permutation gate.
- In characteristic 2, linearity is preserved.
- TOF: controls u_i, u_j stay, target u_k changes to $2u_i u_j u_k - u_i u_j - u_k$.
- Linearity not preserved.

Equality Constraints

To enforce a desired output value z_i on qubit i with final term u_i:

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

Equality Constraints

To enforce a desired output value z_i on qubit i with final term u_i:

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

In characteristic 2, Q_C remains quadratic.

Equality Constraints

To enforce a desired output value z_i on qubit i with final term u_i:

$$\begin{aligned} P_C &\ast= &(1 + 2u_i z_i - u_i - z_i) \\ Q_C &+= &w_j(u_i + z_i - 2u_i z_i). \end{aligned}$$

In characteristic 2, Q_C remains quadratic.

Theorem (Cai-Chen-Lipton-Lu 2010, after Grigoriev-Karpinski; various)

For quadratic $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_K$, and all $a < K$, $N_p[a]$ is computable in $maths f poly(nK)$ time.

Equality Constraints

To enforce a desired output value z_i on qubit i with final term u_i:

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

In characteristic 2, Q_C remains quadratic.

Theorem (Cai-Chen-Lipton-Lu 2010, after Grigoriev-Karpinski; various)

For quadratic $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_K$, and all $a < K$, $N_p[a]$ is computable in $maths f poly(nK)$ time.

Open: replace K by $\log K$ in the time? Affirmative for $A[C(x) \mapsto z]$.

Gottesman-Knill: alternative methodology

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C += 2u_iy_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.
- Equality constraint $w_j(u_i + z_i - 2u_i z_i)$: OK with [G-K], [CCLL] because w_j appears only here.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.
- Equality constraint $w_j(u_i + z_i - 2u_i z_i)$: OK with [G-K], [CCLL] because w_j appears only here.
- S: u_i left alone but $Q_C \mathrel{+}= u_i^2$.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.
- Equality constraint $w_j(u_i + z_i - 2u_i z_i)$: OK with [G-K], [CCLL] because w_j appears only here.
- S: u_i left alone but $Q_C \mathrel{+}= u_i^2$.
- Inductively every term in Q_C has form y_j^2 or $2y_i y_j$.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.
- Equality constraint $w_j(u_i + z_i - 2u_i z_i)$: OK with [G-K], [CCLL] because w_j appears only here.
- S: u_i left alone but $Q_C \mathrel{+}= u_i^2$.
- Inductively every term in Q_C has form y_j^2 or $2y_i y_j$.
- These terms are invariant under $0 \leftrightarrow 2$, $1 \leftrightarrow 3$.

Gottesman-Knill: alternative methodology

- To represent u_i—S— we need $K = 4$.
- H gives $Q_C \mathrel{+}= 2u_i y_j$.
- CNOT: Nonlinear term has a 2 which will cancel the 2 from Hadamard.
- Equality constraint $w_j(u_i + z_i - 2u_i z_i)$: OK with [G-K], [CCLL] because w_j appears only here.
- S: u_i left alone but $Q_C \mathrel{+}= u_i^2$.
- Inductively every term in Q_C has form y_j^2 or $2y_i y_j$.
- These terms are invariant under $0 \leftrightarrow 2$, $1 \leftrightarrow 3$.
- Hence [CCLL] gives poly-time simulation by soution counting in $\mathbb{Z}_4$.

Open Questions

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?
- What else is (physically!) meaningful about polynomials in the circuit's partition function?

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?
- What else is (physically!) meaningful about polynomials in the circuit's partition function?
- Invariants based on Strassen's *geometric degree* $\gamma(f)$ concept, others?

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?
- What else is (physically!) meaningful about polynomials in the circuit's partition function?
- Invariants based on Strassen's *geometric degree* $\gamma(f)$ concept, others?
- Baur-Strassen showed that $\log_2 \gamma(f)$ lower-bounds the arithmetical complexity of f, indeed the number of binary multiplication gates.

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?
- What else is (physically!) meaningful about polynomials in the circuit's partition function?
- Invariants based on Strassen's *geometric degree* $\gamma(f)$ concept, others?
- Baur-Strassen showed that $\log_2 \gamma(f)$ lower-bounds the arithmetical complexity of f, indeed the number of binary multiplication gates.
- Relevance to complexity of quantum C?

Open Questions

- Solution counting is #P-complete for degree 3 over $\mathbb{Z}_K$ in general.
- Are some "structured" subcases of degree 3 tractable? Can they come from families of QC's?
- What else is (physically!) meaningful about polynomials in the circuit's partition function?
- Invariants based on Strassen's *geometric degree* $\gamma(f)$ concept, others?
- Baur-Strassen showed that $\log_2 \gamma(f)$ lower-bounds the arithmetical complexity of f, indeed the number of binary multiplication gates.
- Relevance to complexity of quantum C?
- Possibly quantify the "entangling capacity" of C?