Fast estimation of outcome probabilities for quantum circuits

Kamil Korzekwa

Faculty of Physics, Astronomy and Applied Computer Science, Jagiellonian University, Poland

JAGIELLONIAN UNIVERSITY IN KRAKÓW

FNP Foundation for Polish Science

TEAM-NET

Collaborators

Oliver Reardon-Smith

JAGIELLONIAN UNIVERSITY IN KRAKÓW

Hakop Pashayan

PERIMETER INSTITUTE

UNIVERSITY OF WATERLOO | IQC Institute for Quantum Computing

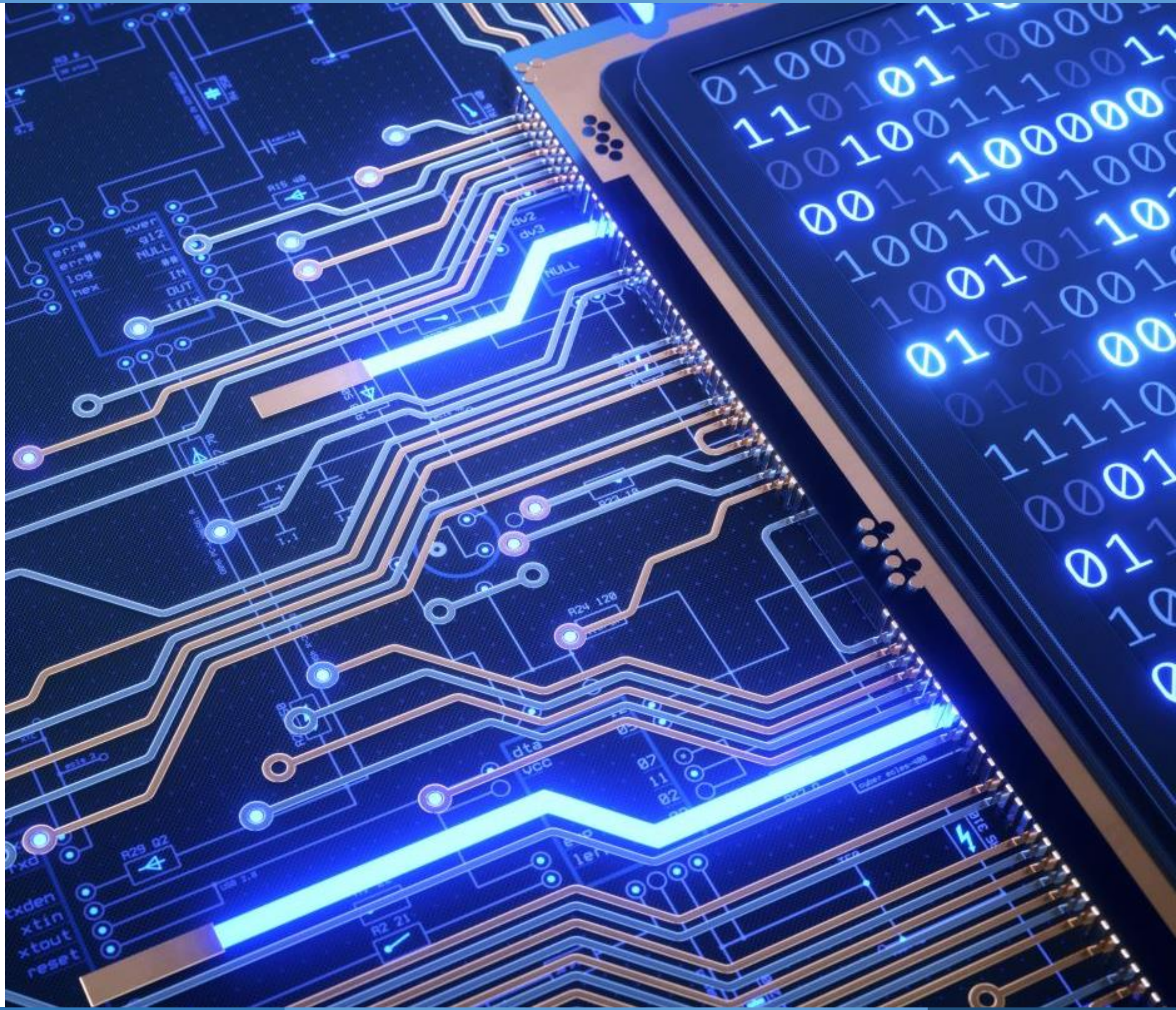
Stephen Bartlett

THE UNIVERSITY OF SYDNEY

Outline

1. Intro
   - Statement of the problem
   - Motivation
2. Algorithms and their performance
   - Overview
   - `COMPRESS` algorithm
   - `COMPUTE` algorithm
   - `RAWESTIM` algorithm
   - `ESTIMATE` algorithm
3. Outlook

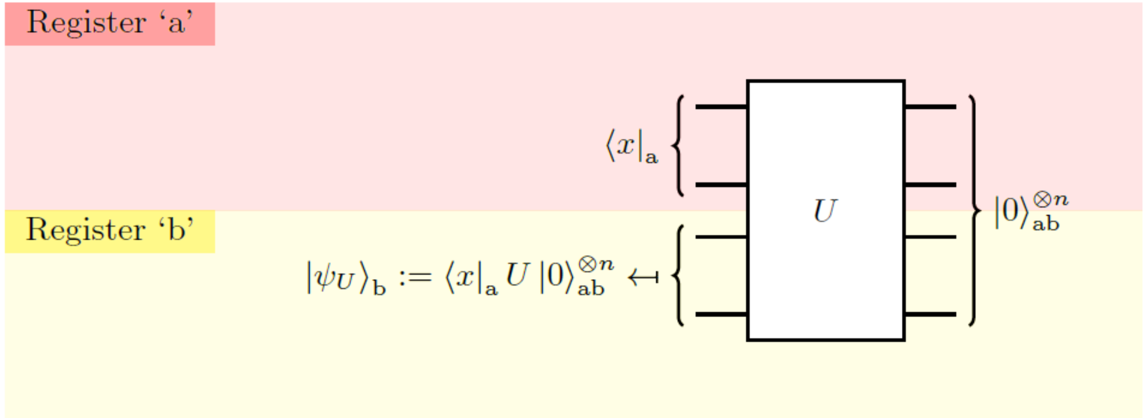
More details:
arXiv:2101.12223
github.com/or1426/Clifford-T-estimator



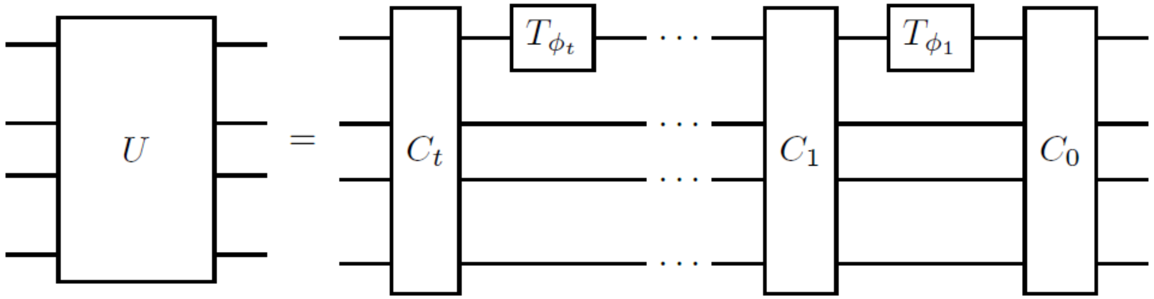
Statement of the problem

Outcome probability for a general quantum circuit with n qubits and w measured qubits:

$$p := \left\| \langle x|_{\mathrm{a}}\, U\, |0\rangle_{\mathrm{ab}}^{\otimes n} \right\|_2^2$$



Elementary description of the circuit:



Clifford gates C_n generated by:

$$S \equiv \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad CX \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

“Magic” gates: $T_\phi \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$

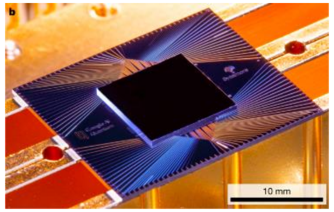
Motivation

Foundations

What components of quantum theory are hard to simulate
and may be responsible for the quantum speed-up?

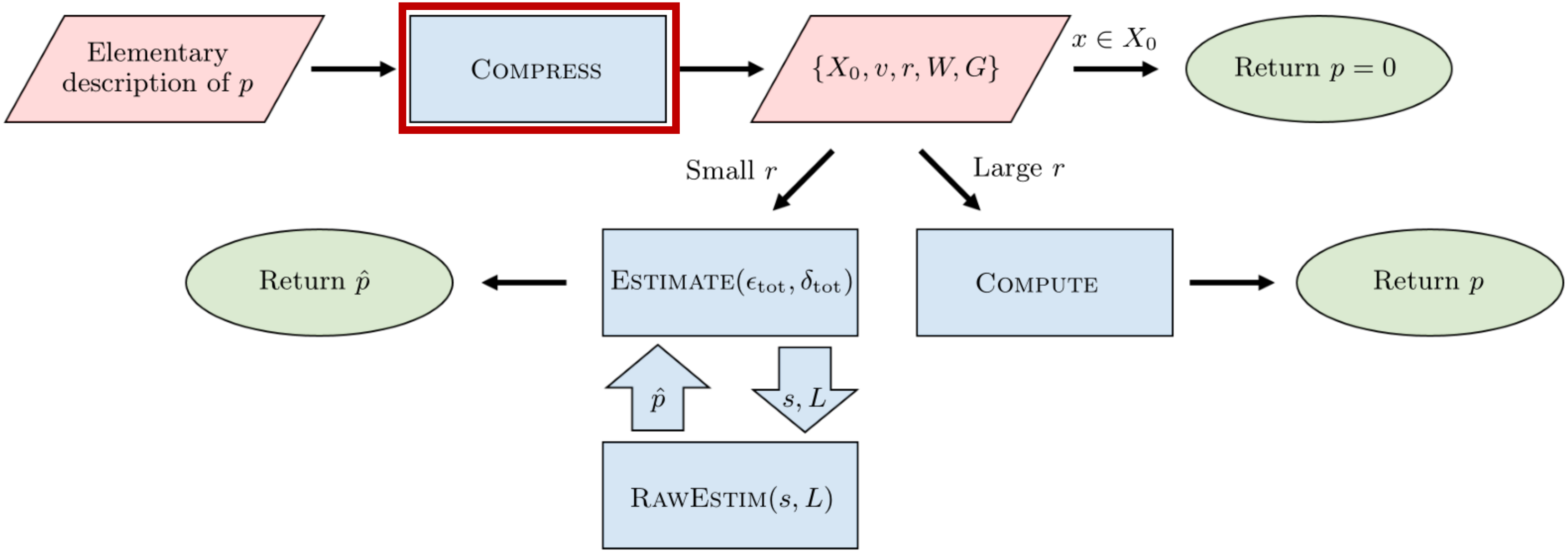
Applications

Characterization, verification, and validation of near-term quantum devices



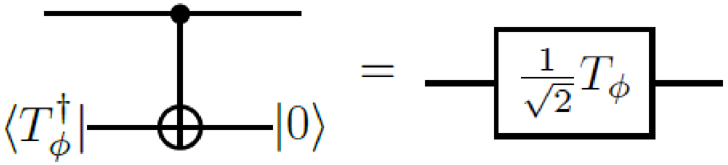
1. Run quantum computer R times
2. Form histogram: R samples over 2^w outcomes
3. Choose $k < 2^w$ outcomes
4. Use our algorithm to estimate these probabilities
5. Compare estimated and measured probabilities

Overview


Elementary description of p
COMPRESS
$\{X_0, v, r, W, G\}$
$x \in X_0$
Return $p = 0$
Small r
Large r
Return $\hat{p}$
ESTIMATE$(\epsilon_{\text{tot}}, \delta_{\text{tot}})$
COMPUTE
Return p
$\hat{p}$
s, L
RAWESTIM(s, L)

COMPRESS algorithm

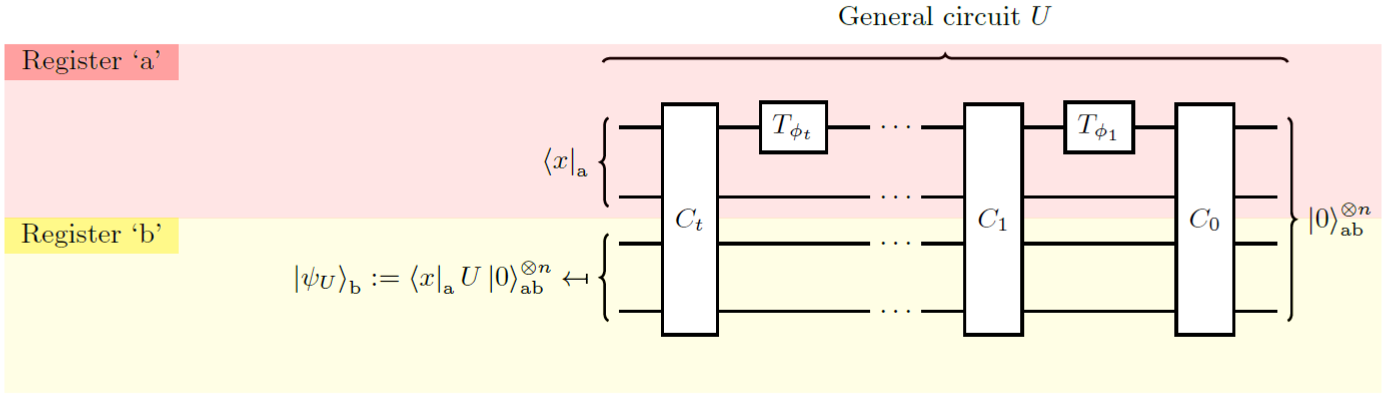
Step 1: Gadgetization

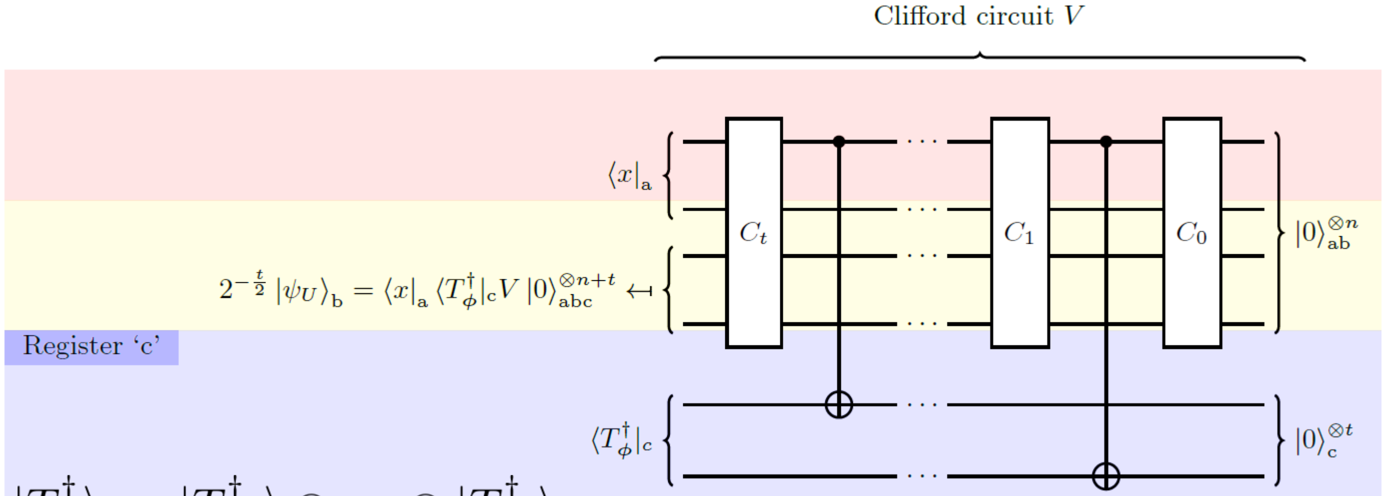


$$|T_\phi^\dagger\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \exp(-i\phi)\,|1\rangle)$$

Replace $\quad p = \left\| \langle x|_{\mathrm{a}}\, U\, |0\rangle_{\mathrm{ab}}^{\otimes n} \right\|_2^2$

With: $\quad p = 2^t \left\| \langle x|_{\mathrm{a}}\, \langle T_\phi^\dagger|_{\mathrm{c}} V\, |0\rangle_{\mathrm{abc}}^{\otimes n+t} \right\|_2^2$





$$|T_\phi^\dagger\rangle := |T_{\phi_1}^\dagger\rangle \otimes \cdots \otimes |T_{\phi_t}^\dagger\rangle$$

COMPRESS algorithm

Step 2: Constrain stabilizers

$$p = 2^t \left\| \langle x|_{\mathrm{a}} \langle T_\phi^\dagger|_{\mathrm{c}} V |0\rangle_{\mathrm{abc}}^{\otimes n+t} \right\|_2^2 = 2^t \mathrm{Tr}\left(\boxed{\mathrm{Tr}_{\mathrm{ab}}(V |0\rangle\langle 0|_{\mathrm{abc}}^{\otimes n+t} V^\dagger |x\rangle\langle x|_{\mathrm{a}})} |T_\phi^\dagger\rangle\langle T_\phi^\dagger|_{\mathrm{c}} \right)$$

Use stabilizer description: $V |0\rangle\langle 0|_{\mathrm{abc}}^{\otimes n+t} V^\dagger = \Pi_{G^{(0)}}$ $\quad \Pi_{G^{(0)}} := \prod_{i=1}^{n+t} \frac{I + g_i}{2} = 2^{-(n+t)} \sum_{g \in \langle G^{(0)} \rangle} g$

Remove vanishing terms: $\mathrm{Tr}_{\mathrm{ab}}\left(V |0\rangle\langle 0|_{\mathrm{abc}}^{\otimes n+t} V^\dagger |x\rangle\langle x|_{\mathrm{a}}\right) = 2^{-(n+t)} \sum_{g \in \langle G^{(0)} \rangle} \omega(g) \mathrm{Tr}\left(|g|_{\mathrm{a}} |x\rangle\langle x|_{\mathrm{a}}\right) \mathrm{Tr}\left(|g|_{\mathrm{b}}\right) |g|_{\mathrm{c}}$

- Register ‘a’ constraints: for all $j \in [w]$, $|g|_j \in \{I, Z\}$
- Register ‘b’ constraints: for all $j \in [n-w]$, $|g|_{w+j} = I$.

COMPRESS algorithm

Step 2: Constrain stabilizers - continued

After a lenghty stabilizer analysis:

$$\mathrm{Tr}_{\mathrm{ab}}\left(V\,|0\rangle\langle 0|_{\mathrm{abc}}^{\otimes n+t}\,V^\dagger\,|x\rangle\langle x|_{\mathrm{a}}\right) = 2^{-r+v-w}\Pi_G$$

G - stabilizer group on t qubits with $t-r$ stabilisers

And so the outcome probability:

$$p = 2^{t-r+v-w}\mathrm{Tr}\left(\Pi_G|T_\phi^\dagger\rangle\langle T_\phi^\dagger|\right)$$

Step 3: Gate sequence construction

Use destabiliser+stabiliser tableaux of Aaronson and Gottesman* to find explicit construction of a Clifford unitary *W:*

$$\Pi_G = W^\dagger(|0\rangle\langle 0|^{\otimes t-r}\otimes I^{\otimes r})W.$$

And so the outcome probability:

$$p = 2^{t-r+v-w}\left\|\langle 0|^{\otimes t-r}\,W|T_\phi^\dagger\rangle\right\|_2^2$$

*S. Aaronson and D. Gottesman, Improved simulation of stabilizer circuits, Phys. Rev. A **70**, 052328 (2004).

COMPRESS algorithm

Bottom line:

In polynomial time COMPRESS obtains two expressions for the outcome probability:

$$p = \boxed{2^{t-r+v-w} \left\| \langle 0|^{\otimes t-r} W |T_\phi^\dagger\rangle \right\|_2^2} = \boxed{2^{v-w} \langle T_\phi^\dagger| \prod_{i=1}^{t-r} (I + g_i) |T_\phi^\dagger\rangle}$$

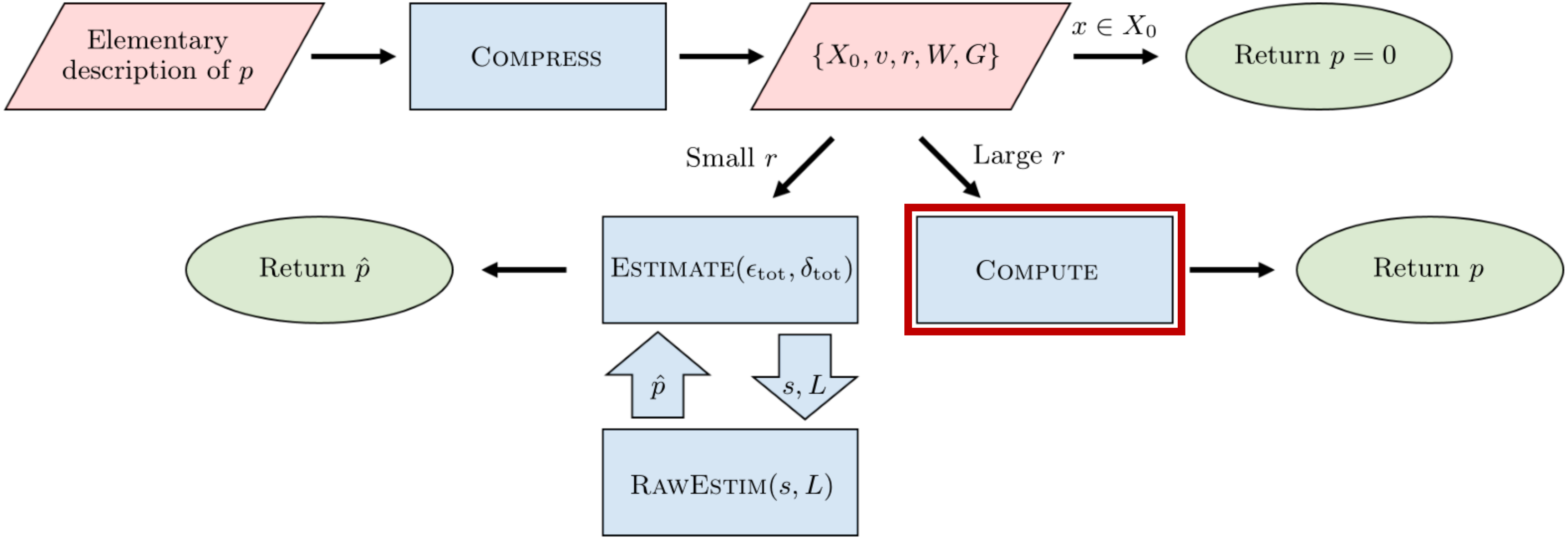
To be used by the RAWESTIM algorithm

To be used by the COMPUTE algorithm

$$r \in \{0, 1, \ldots, \min\ \{t, n-w\}\}$$

It also specifies a size v subset of the measured qubits that have deterministic outcomes and provides the measurement outcomes these qubits must produce.

Overview


Elementary description of p
COMPRESS
$\{X_0, v, r, W, G\}$
$x \in X_0$
Return $p = 0$
Small r
Large r
Return $\hat{p}$
ESTIMATE$(\epsilon_{\text{tot}}, \delta_{\text{tot}})$
COMPUTE
Return p
$\hat{p}$
s, L
RAWESTIM(s, L)

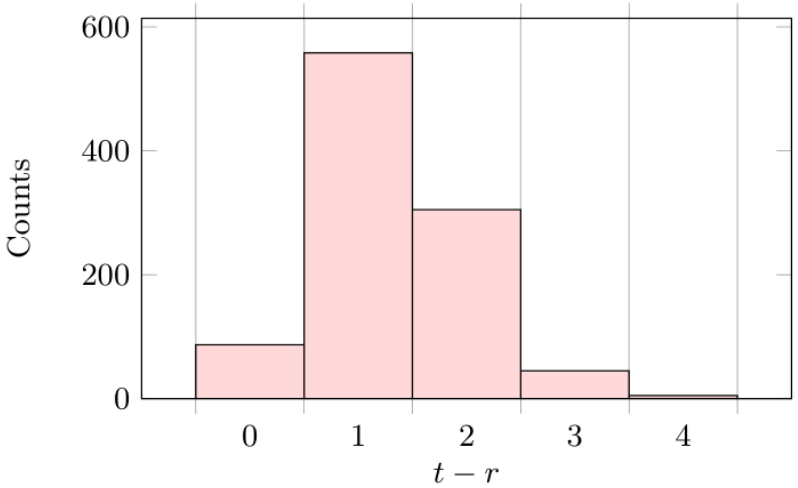
COMPUTE algorithm

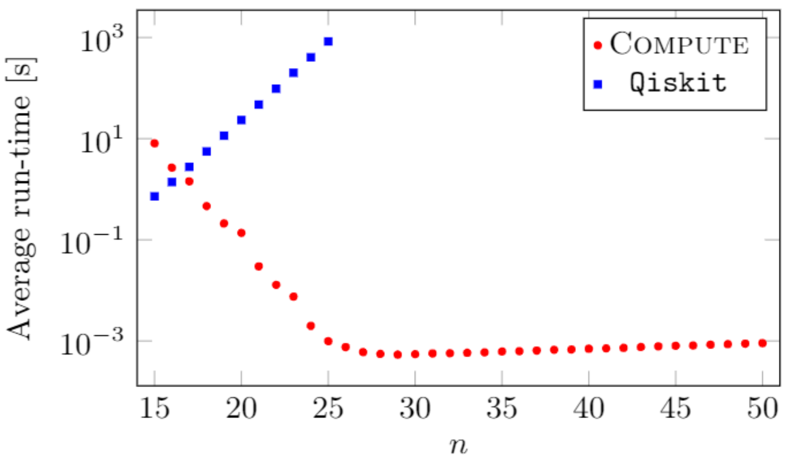
COMPUTE exactly computes p in $\tau = O(2^{t-r}(t-r)t)$:

$$p = 2^{v-w} \langle T_\phi^\dagger | \prod_{i=1}^{t-r} (I + g_i) | T_\phi^\dagger \rangle$$

$r \in \{0, 1, \ldots, \min\{t, n-w\}\}$ concentrates near the maximum for large c random circuits

10^3 RCs
$n = 100$
$w = 20$
$t = 80$
$c = 10^5$

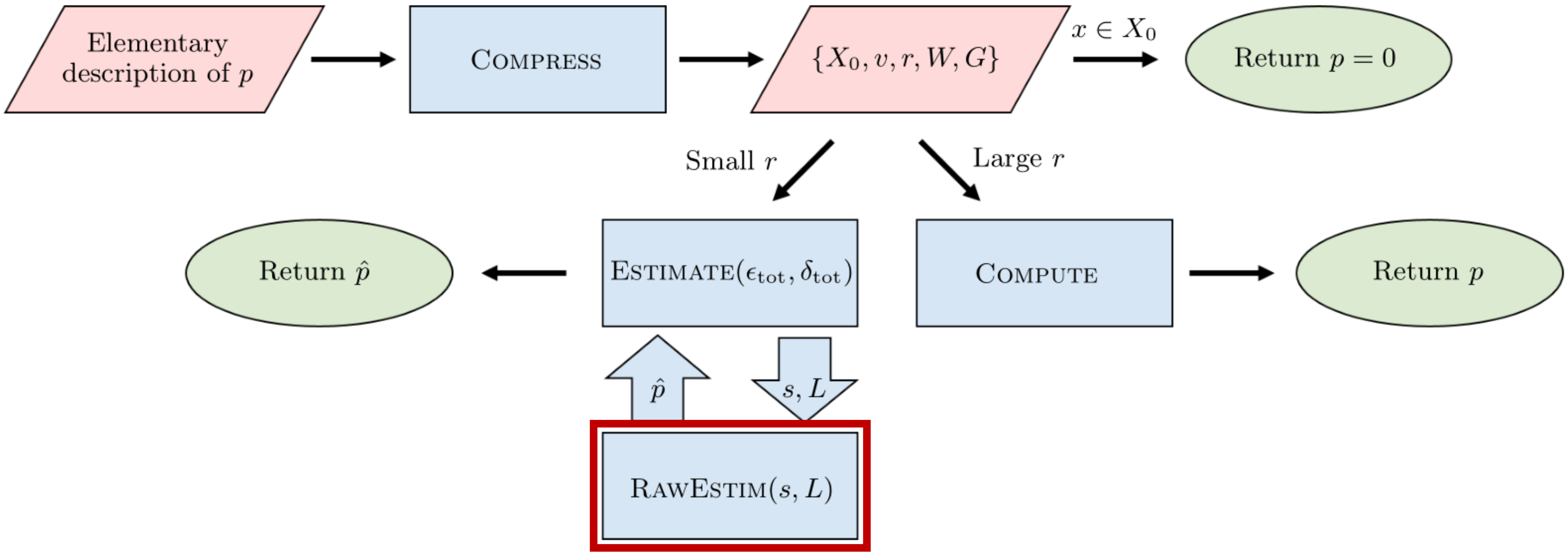




$\boxed{n = 55,\ w = 5,\ t = 80,\ c = 10^5, \text{ runtime} \leq 2 \text{ hours}}$

$w = 10,\ t = 30,\ c = 10^3$, averaged over 100 circuits

Overview



Elementary description of p
COMPRESS
$\{X_0, v, r, W, G\}$
$x \in X_0$
Return $p = 0$
Small r
Large r
Return $\hat{p}$
ESTIMATE$(\epsilon_{\text{tot}}, \delta_{\text{tot}})$
COMPUTE
Return p
$\hat{p}$
s, L
RAWESTIM(s, L)

RAWESTIM algorithm

$$\boxed{p = 2^{t-r+v-w} \left\| \langle 0|^{\otimes t-r} \, W |T_{\phi}^{\dagger}\rangle \right\|_2^2}$$

Step 1: Stabiliser decomposition and sampling

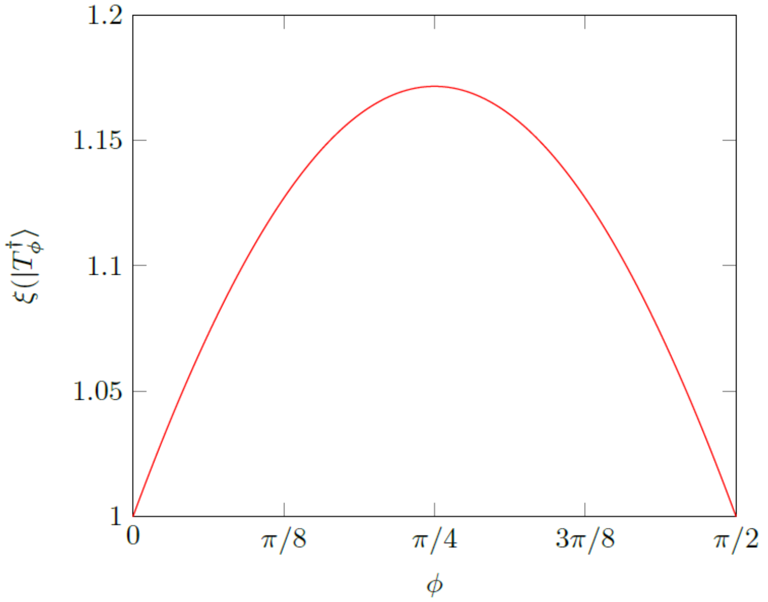
Introduce non-standard notation: $\left|\tilde{0}\right\rangle := \left|+\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle), \qquad \left|\tilde{1}\right\rangle := \left|-i\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle - i\left|1\right\rangle)$

Decompose each magic state as: $\left|T_{\phi_j}^{\dagger}\right\rangle = \alpha_{\phi_j}\left|\tilde{0}\right\rangle + \alpha'_{\phi_j}\left|\tilde{1}\right\rangle$

This decomposition minimises the stabilizer extent:

$$\xi(\left|\psi\right\rangle) := \min_c \left\{ \|c\|_1^2 \;\middle|\; \left|\psi\right\rangle = \sum_j c_j \left|\sigma_j\right\rangle, \quad \left|\sigma_j\right\rangle \text{ is a stabiliser state} \right\}$$

Define total stabilizer extent: $\xi^* := \xi(|T_{\phi}^{\dagger}\rangle) = \prod_{j=1}^{t} \xi(|T_{\phi_j}^{\dagger}\rangle)$



RAWESTIM algorithm

Step 1: Stabiliser decomposition and sampling - continued

Use the decomposition to find:

$$p = \xi^* \cdot 2^{t-r+v-w} \left\| \sum_y q(y) \prod_{j=1}^{t} e^{i\varphi_j(1-y_j)} e^{i\varphi'_j y_j} \langle 0|^{\otimes t-r} W |\tilde{y}\rangle \right\|_2^2$$

With product probability distribution:

$$q(y) = \prod_{j=1}^{t} q(y_j), \quad q(y_j) = \begin{cases} \frac{|\alpha_{\phi_j}|}{|\alpha_{\phi_j}|+|\alpha'_{\phi_j}|} & \text{for } y_j = 0 \\ \frac{|\alpha'_{\phi_j}|}{|\alpha_{\phi_j}|+|\alpha'_{\phi_j}|} & \text{for } y_j = 1 \end{cases}$$

Thus, introducing:

$$|\psi(y)\rangle := \sqrt{\xi^*} \cdot 2^{\frac{t-r+v-w}{2}} \prod_{j=1}^{t} e^{i\varphi_j(1-y_j)} e^{i\varphi'_j y_j} \langle 0|^{\otimes t-r} W |\tilde{y}\rangle$$

We have:

$$p = \| |\mu\rangle \|_2^2, \qquad |\mu\rangle := \mathop{\mathbb{E}}_{Y\sim q} [|\psi(Y)\rangle] = \sum_y q(y) |\psi(y)\rangle$$

RAWESTIM algorithm

Step 1: Stabiliser decomposition and sampling - continued

Instead of the mean vector $|\mu\rangle$ we use s-sample average: $|\overline{\psi}\rangle = \frac{1}{s}\sum_{j=1}^{s} |\psi_{x_j}\rangle$

Our crucial concentration result based on generalised Hoeffding inequality*:

$$\Pr\left(\left|\||\overline{\psi}\rangle\|_2^2 - p\right| \geq \epsilon\right) \leq \delta, \quad \delta := 2e^2 \exp\left(\frac{-s(\sqrt{p+\epsilon} - \sqrt{p})^2}{2(\sqrt{\xi^*} + \sqrt{p})^2}\right)$$

$$p = \||\mu\rangle\|_2^2$$

So how do we get $|\overline{\psi}\rangle$? And how do we calculate $\||\overline{\psi}\rangle\|_2^2$?

*T. P. Hayes, A large-deviation inequality for vector-valued martingales, Combinatorics, Probability and Computing (2005).

RAWESTIM algorithm

Step 2: State evolution

Use CH-form*, the phase-sensitive generalisation of Gottesman-Knill theorem:

$$|\psi(y)\rangle := \sqrt{\xi^*} \cdot 2^{\frac{t-r+v-w}{2}} \prod_{j=1}^{t} e^{i\varphi_j(1-y_j)} e^{i\varphi'_j y_j} \langle 0|^{\otimes t-r} W |\tilde{y}\rangle$$

*S. Bravyi, *et al.*, Simulation of quantum circuits by low-rank stabilizer decompositions, Quantum **3**, 181 (2019).

Step 3: Fast norm estimation

Estimate the norm of $|\overline{\psi}\rangle = \frac{1}{s}\sum_{j=1}^{s} |\psi_{x_j}\rangle$ from overlaps of $|\psi_{x_j}\rangle$ with L random stabiliser states

*S. Bravyi and D. Gosset, Improved classical simulation of quantum circuits dominated by Clifford gates, Phys. Rev. Lett. **116**, 250501 (2016)

RAWESTIM algorithm

Bottom line:

For all $\epsilon_{\text{tot}} > 0$ and $\epsilon \in (0, \epsilon_{\text{tot}})$ we get an estimate:

$$\boxed{\Pr\left(|\hat{p} - p| \geq \epsilon_{\text{tot}}\right) \leq 2e^2 \exp\left(\frac{-s(\sqrt{p+\epsilon} - \sqrt{p})^2}{2(\sqrt{\xi^*} + \sqrt{p})^2}\right) + \exp\left(-\left(\frac{\epsilon_{\text{tot}} - \epsilon}{p+\epsilon}\right)^2 L\right) =: \delta_{\text{tot}}}$$

With runtime $\tau = O(st^2(t-r) + sLr^3)$

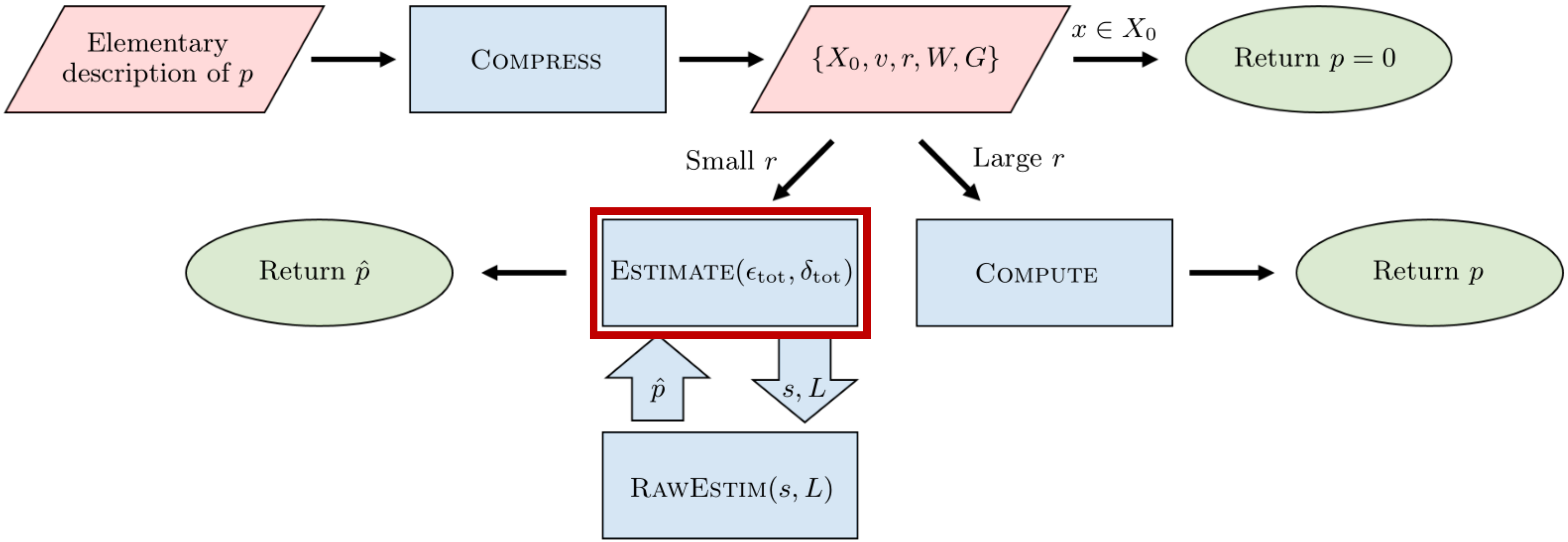
Alternatively:

We get an estimate: $\Pr\left(|\hat{p} - p| \geq \epsilon_{\text{tot}}\right) \leq \delta_{\text{tot}}$.

Whenever:

$$s \geq \frac{2(\sqrt{\xi^*} + \sqrt{p})^2}{\left(\sqrt{p+\epsilon} - \sqrt{p}\right)^2} \log\left(\frac{2e^2}{\delta}\right), \qquad L \geq \left(\frac{p+\epsilon}{\epsilon_{\text{tot}} - \epsilon}\right)^2 \log\left(\frac{1}{\delta_{\text{tot}} - \delta}\right).$$

Overview

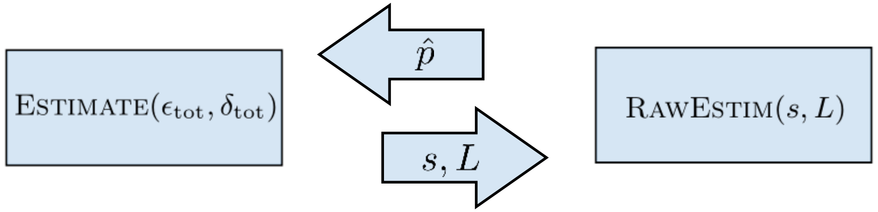


Elementary description of p
COMPRESS
$\{X_0, v, r, W, G\}$
$x \in X_0$
Return $p = 0$
Small r
Large r
Return $\hat{p}$
ESTIMATE($\epsilon_{\mathrm{tot}}, \delta_{\mathrm{tot}}$)
COMPUTE
Return p
$\hat{p}$
s, L
RAWESTIM(s, L)

ESTIMATE algorithm

ESTIMATE purpose: $\Pr\left(|\hat{p} - p| \geq \epsilon_{\text{tot}}\right) \leq \delta_{\text{tot}}$

ESTIMATE uses RAWESTIM as a subroutine:



RAWESTIM gives: $\Pr\left(|\hat{p} - p| \geq \epsilon_{\text{tot}}\right) \leq \delta(p, \epsilon_{\text{tot}}, \epsilon, s, L)$

RAWESTIM's runtime cost: $\tau(s, L) := st^2(t - r) + sLr^3$

Define $\epsilon^*(p, \delta_{\text{tot}}, \mathcal{T})$ as the minimal achievable error s.t. $\tau(s, L) < \mathcal{T}$ and failure probability $< \delta_{\text{tot}}$

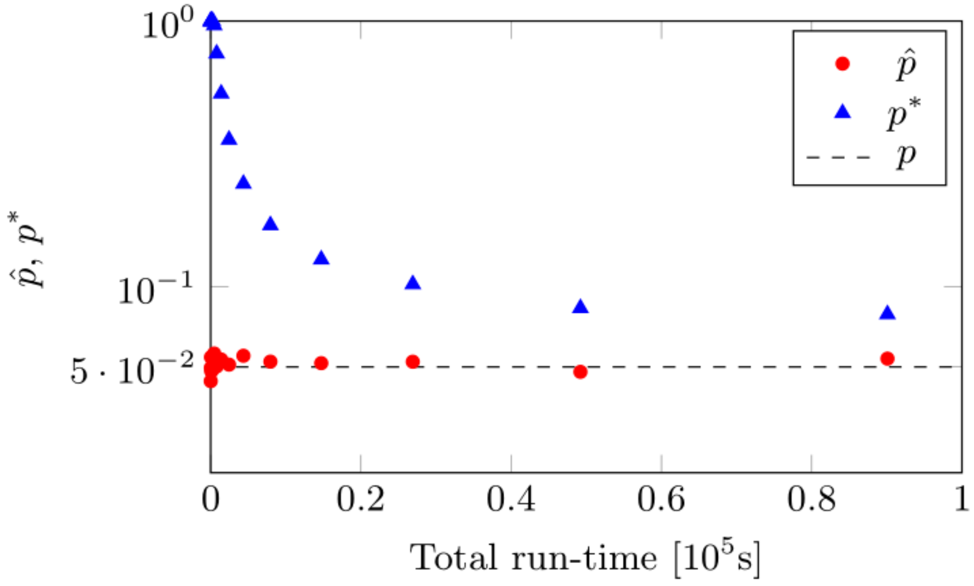
We prove that $\epsilon^*(p, \delta_{\text{tot}}, \mathcal{T})$ is monotonically increasing in p

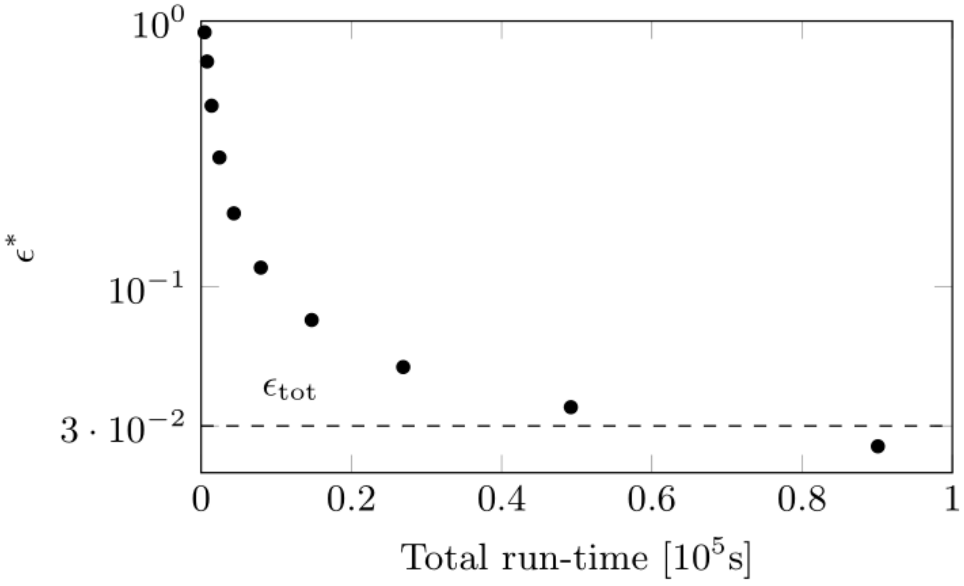
ESTIMATE algorithm

Idea: for $k = 1, 2, \ldots$, compute $\epsilon_k^* := \epsilon^*(p_{k-1}^*, \delta_k, 2^k \mathcal{T}_0)$ until we find $\epsilon^*(p^*, \delta_k, 2^k \mathcal{T}_0) \leq \epsilon_{\mathrm{tot}}$

$p_0^* = 1$, $p_k^* = \hat{p}_k + \epsilon_k^*$ and

$\delta_k := \frac{6}{\pi^2 k^2} \delta_{\mathrm{tot}}$ ensures $\sum_k \delta_k = \delta_{\mathrm{tot}}$



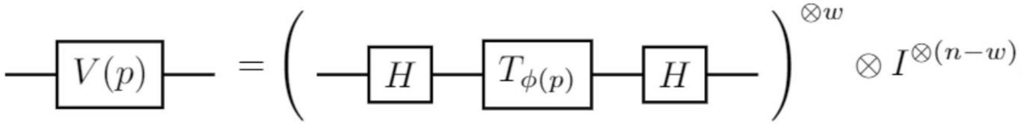


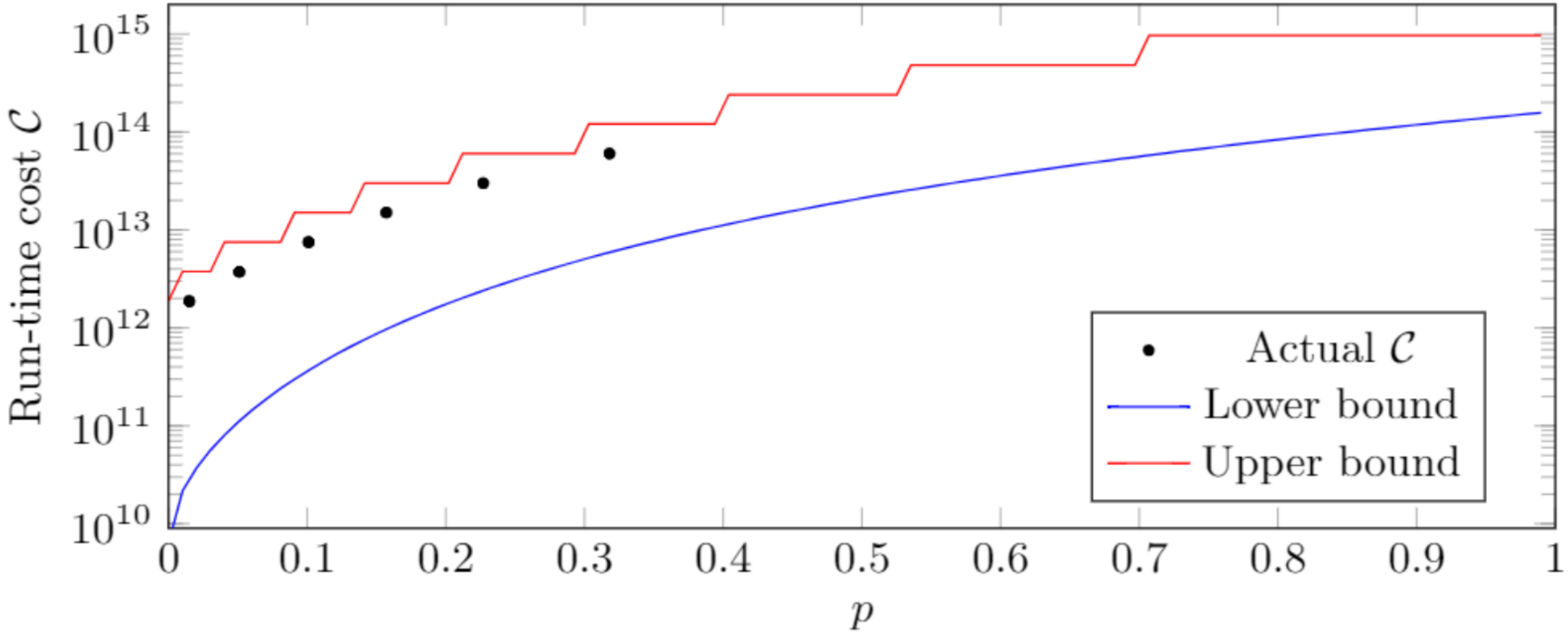
$$n = 50,\ w = 8,\ t = 60,\ c = 10^3,\ \delta_{\mathrm{tot}} = 10^{-3}$$

ESTIMATE algorithm

ESTIMATE run-time as a function of p

($\epsilon_{\mathrm{tot}} = 0.05,\ \delta_{\mathrm{tot}} = 10^{-3}$ for random circuits $UU^{\dagger}V(p)$)





Conclusion: RAWESTIM's runtime with optimal choice of paramaters can be used as a proxy for ESTIMATE runtime (up to 2 orders of magnitude).

Outlook

1. Applications
   - Can we design useful quantum verification schemes employing outcome estimation algorithms?
   - Can we use it to verify toy quantum computers?
2. Extensions
   - Can we go from Clifford+T paradigm to matchgates+SWAP?
   - Can we generalise to arbitrary free subtheory + magic gate?
3. Improvements
   - Can we adapt our algorithms to mixed state formalism?

More details:
arXiv:2101.12223
github.com/or1426/Clifford-T-estimator

Thank you!