Classical simulation complexity of extended Clifford circuits

Quantum and Linear Optical Computation (QLOC) | Journal Club

F. C. R. Peres | 24th of March 2021

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

Presentation overview

1. Introductory concepts
   - ➡ Strong vs. Weak simulation
   - ➡ The Pauli group
   - ➡ The Clifford group and stabiliser circuits
   - ➡ The stabiliser formalism and the Gottesman-Knill theorem

Presentation overview

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

2. Classical simulation complexity of extended Clifford circuit - paper review[*]

➡ The extra ingredients

➡ The extended classes

➡ Theorems and proofs

[*] R. Jozsa and M. V. den Nest, Quantum Information and Computation **14** (2013), arXiv:1305.6190.

Introductory concepts

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

➡ Strong vs. Weak simulation

- Strong simulation → calculate the probability of any desired outcome of the computation.

- Weak simulation → sample from the output distribution of the circuit.

Introductory concepts

➡ Strong vs. Weak simulation

- Strong simulation → calculate the probability of any desired outcome of the computation.

- Weak simulation → sample from the output distribution of the circuit.

- **Lemma 1:** If a given circuit can be efficiently classically simulated in the strong sense, then it can also be efficiently classically simulated in the weak sense.

Introductory concepts

➡ Strong vs. Weak simulation

- Strong simulation → calculate the probability of any desired outcome of the computation.

- Weak simulation → sample from the output distribution of the circuit.

- **Lemma 1:** If a given circuit can be efficiently classically simulated in the strong sense, then it can also be efficiently classically simulated in the weak sense.

Introductory concepts

➡ The Pauli group

- **Definition 1:** [PAULI GROUP]

  The Pauli group on N qubits $\mathscr{P}_N$ is the group whose elements are N-fold tensor products of the single-qubit Pauli operators I, X, Y and Z, together with the multiplicative factors ± 1 and $\pm i$.

- Number of elements : 4^{N+1}; $N = 1 \Rightarrow 16$ elements; $N = 2 \Rightarrow 64$ elements.

- **Example:** $(X \otimes I), (X \otimes X), -i(Y \otimes Z) \in \mathscr{P}_2$

Introductory concepts

➡ The Pauli group

- $\mathscr{P}_N$ can be completely described by $2N$ generators:

$$\mathscr{P}_N = \langle X_1, \ldots, X_N, Z_1, \ldots, Z_N \rangle .$$

- **Notation:** X_i denotes the operator such that the single-qubit Pauli X acts on the i-th qubit of the system and the identity is applied to all other qubits. Example: $X_1 = (X \otimes I \otimes \ldots \otimes I) = (X_{(1)} \otimes I_{(2)} \otimes \ldots \otimes I_{(N)})$.

Introductory concepts

➡ The Pauli group

- **Example:** $\mathscr{P}_2$ has 64 elements but only 4 generators:

$$\mathscr{P}_2 = \langle X \otimes I,\, I \otimes X,\, Z \otimes I,\, I \otimes Z \rangle = \left\langle X_1,\, X_2,\, Z_1,\, Z_2 \right\rangle .$$

- **Example:** $\mathscr{P}_3$ has 256 elements but only 6 generators:

$$\mathscr{P}_3 = \left\langle X_1,\, X_2,\, X_3,\, Z_1,\, Z_2,\, Z_3 \right\rangle .$$

Introductory concepts

➡ The Clifford group and stabiliser circuits

- **Definition 2:** [CLIFFORD GROUP]

  An operation is said to be a Clifford unitary C if it maps the Pauli group onto itself under conjugation, that is, if

$$C\mathscr{P}_N C^\dagger = \mathscr{P}_N \Leftrightarrow CP_iC^\dagger = P_j \,,$$

  where $P_i, P_j \in \mathscr{P}_N$.

  *Clifford unitaries form a group known as the Clifford group and generated by the Hadamard (H), phase (S) and controlled-*NOT *(CX) gates.*

Introductory concepts

➡ The Clifford group and stabiliser circuits

- Action of the generators of the Clifford group on the Pauli group generators:

$$HXH^\dagger = Z; \quad HZH^\dagger = X;$$

$$SXS^\dagger = Y; \quad SZS^\dagger = Z;$$

$$CX(X \otimes I)CX^\dagger = (X \otimes X); \quad CX(I \otimes X)CX^\dagger = (I \otimes X);$$

$$CX(Z \otimes I)CX^\dagger = (Z \otimes I); \quad CX(I \otimes Z)CX^\dagger = (Z \otimes Z).$$

Introductory concepts

➡ The Clifford group and stabiliser circuits

- **Definition 3:** [STABILISER CIRCUIT]

  A circuit is said to be a stabiliser circuit if the following conditions are met:

  (i) its inputs are computational basis states;

  (ii) each operation is either a Clifford unitary or a measurement in the computational basis.

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- **Definition 4:** [STABILISING OPERATION]

  An operator S *is said to stabilise* $|\psi\rangle$ *if:*

$$S\,|\psi\rangle = |\psi\rangle\,.$$

- The stabiliser formalism is a particularly powerful framework for describing stabiliser circuits → in this case the **stabiliser operators are always hermitian Pauli operators**.

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- **Definition 5:** [STABILISER]

  The set of operators P_i which stabilise an N-qubit state $|\psi\rangle$ form a group known as the stabiliser:

$$\mathcal{S} = \{P_i : P_i|\psi\rangle = |\psi\rangle \ \forall P_i \in \mathcal{P}_N\}.$$

- The stabiliser is uniquely determined by N generators.

Introductory concepts

➡ **The stabiliser formalism and the Gottesman-Knill theorem**

- **Example:** $\left|00\right\rangle$

$N = 2 \Rightarrow 2$ generators for the stabiliser: $\mathcal{S} = \langle Z \otimes I, I \otimes Z \rangle$.

- **Example:** Consider the Bell state

$$\left|\mathcal{B}_{00}\right\rangle = \frac{\left|00\right\rangle + \left|11\right\rangle}{\sqrt{2}} .$$

$N = 2 \Rightarrow 2$ generators for the stabiliser: $\mathcal{S} = \langle X \otimes X, Z \otimes Z \rangle$.

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- Schrödinger's picture of quantum mechanics: $\left|\psi\right\rangle \to \left|\psi'\right\rangle = U\left|\psi\right\rangle$.

- Alternatively, we can use Heisenberg's picture.

- In that case, we can describe the evolution of the state through the evolution of its stabiliser:

$$\mathcal{S} \to \mathcal{S}' = U\mathcal{S}U^{\dagger} \, .$$

Introductory concepts

➡ **The stabiliser formalism and the Gottesman-Knill theorem**

- The tableau representation

$$\left(\begin{array}{ccc|ccc|c} x_{11} & \cdots & x_{1N} & z_{11} & \cdots & z_{1N} & s_1 \\ x_{21} & \cdots & x_{2N} & z_{21} & \cdots & z_{2N} & s_2 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{N1} & \cdots & x_{NN} & z_{N1} & \cdots & z_{NN} & s_N \end{array}\right)$$

Introductory concepts

➡ **The stabiliser formalism and the Gottesman-Knill theorem**

- **Example:** Bell state $\left|00\right\rangle$ has stabiliser $\mathcal{S} = \langle Z \otimes I, I \otimes Z \rangle$.

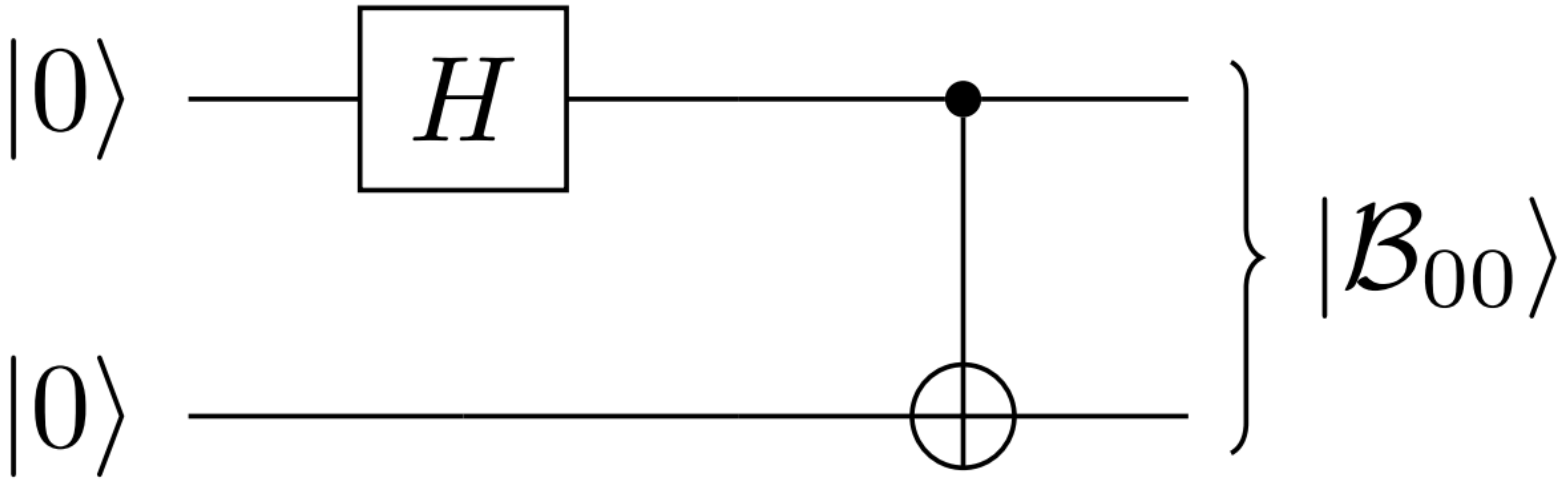
$$\left(\begin{array}{cc|cc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right).$$

- **Example:** Bell state $\left|\mathcal{B}_{00}\right\rangle$ has stabiliser $\mathcal{S} = \langle X \otimes X, Z \otimes Z \rangle$.

$$\left(\begin{array}{cc|cc|c} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Introductory concepts

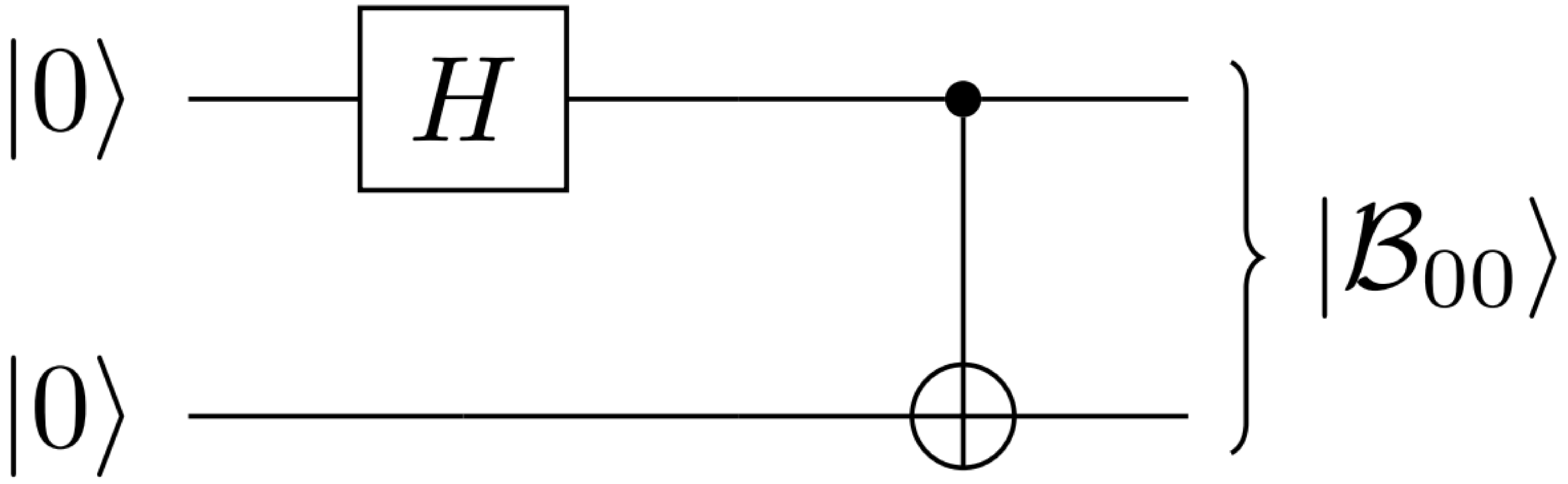
➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \to$$

Introductory concepts

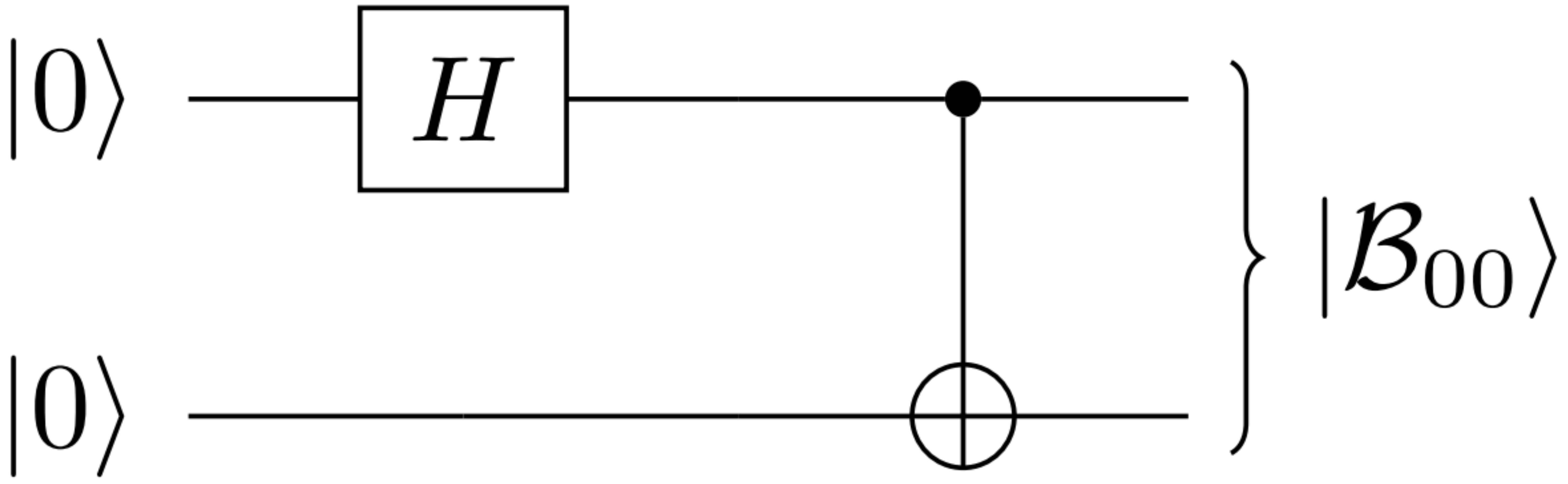
➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} 0 & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow$$

Introductory concepts

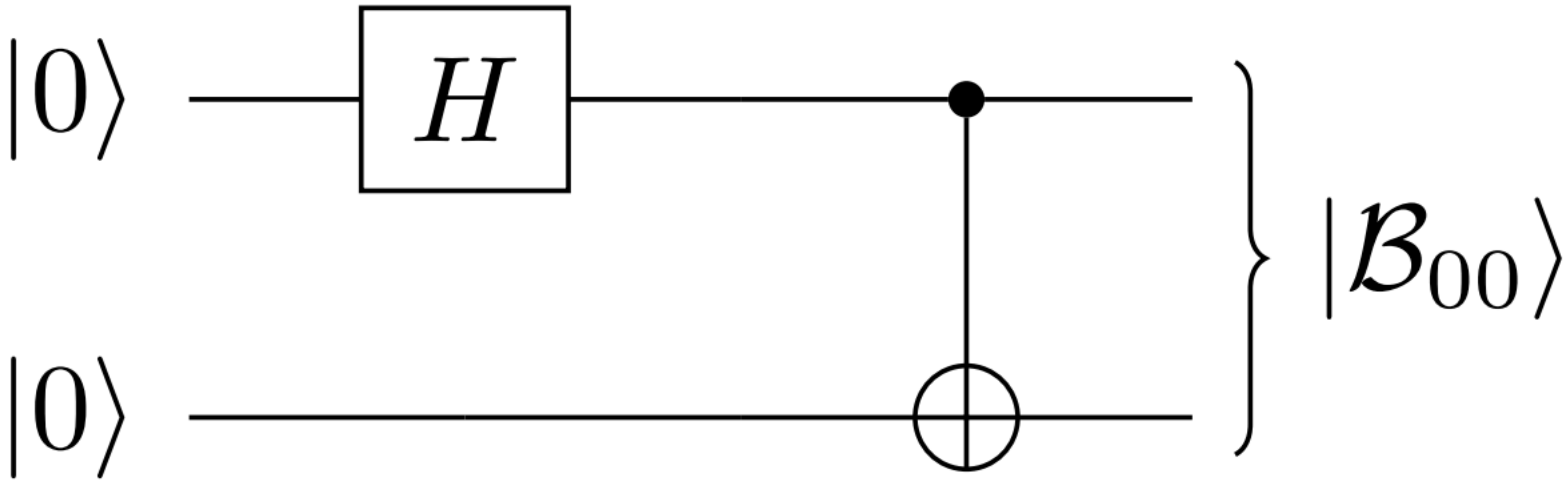
➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} \boxed{0} & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \to$$

Introductory concepts

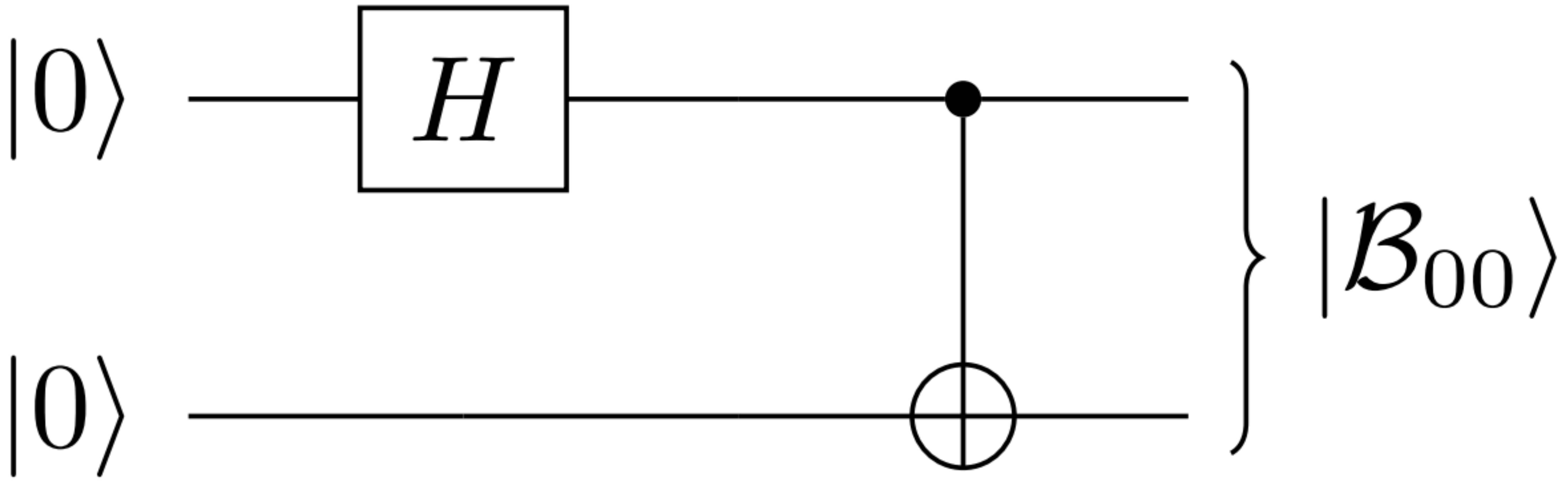
➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} \boxed{0} & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow \left(\begin{array}{cc|cc|c} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow$$

Introductory concepts

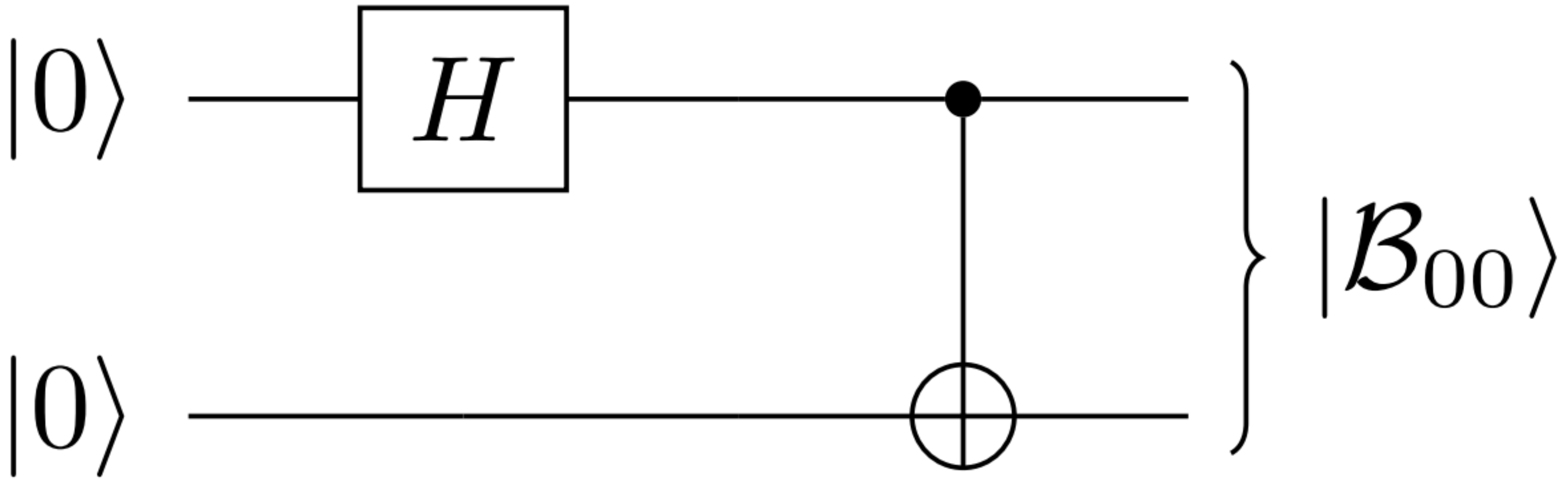
➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} \boxed{0} & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow \left(\begin{array}{cc|cc|c} 1 & \boxed{0} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow$$

Introductory concepts

➡ **The stabiliser formalism and the Gottesman-Knill theorem**

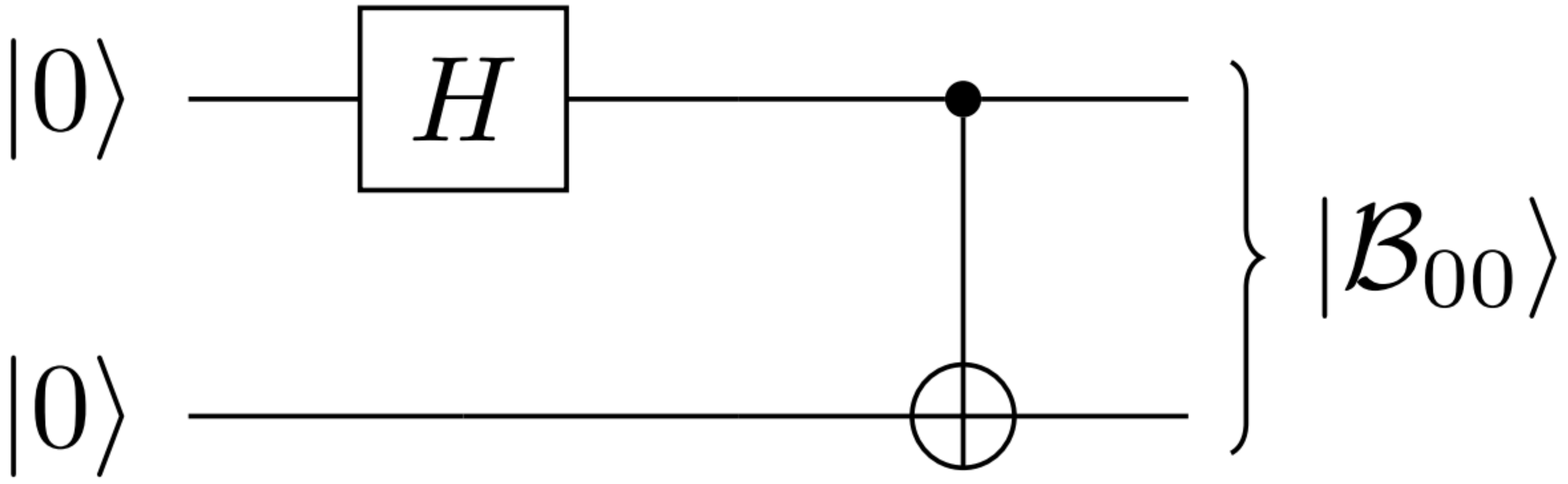


$$\left(\begin{array}{cc|cc|c} \boxed{0} & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \rightarrow \left(\begin{array}{cc|cc|c} 1 & \boxed{0} & 0 & 0 & 0 \\ 0 & 0 & \boxed{0} & 1 & 0 \end{array}\right) \rightarrow$$

Introductory concepts

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

➡ **The stabiliser formalism and the Gottesman-Knill theorem**



$$\left(\begin{array}{cc|cc|c} \boxed{0} & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right) \to \left(\begin{array}{cc|cc|c} 1 & \boxed{0} & 0 & 0 & 0 \\ 0 & 0 & \boxed{0} & 1 & 0 \end{array}\right) \to \left(\begin{array}{cc|cc|c} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Introductory concepts

➡ **The stabiliser formalism and the Gottesman-Knill theorem**

- This formalism provides us with an **efficient** way of tracking the evolution of the state in a stabiliser circuit.

- At each step, the Pauli operators that generate the stabiliser can be updated efficiently through the application of the conjugation rules.

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- This formalism provides us with an **efficient** way of tracking the evolution of the state in a stabiliser circuit.

- At each step, the Pauli operators that generate the stabiliser can be updated efficiently through the application of the conjugation rules.

Efficient way of simulating stabiliser circuits!

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- This formalism provides us with an **efficient** way of tracking the evolution of the state in a stabiliser circuit.

- At each step, the Pauli operators that generate the stabiliser can be updated efficiently through the application of the conjugation rules.

Efficient way of simulating stabiliser circuits!

↓

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- This formalism provides us with an **efficient** way of tracking the evolution of the state in a stabiliser circuit.

- At each step, the Pauli operators that generate the stabiliser can be updated efficiently through the application of the conjugation rules.

Efficient way of simulating stabiliser circuits!

↓

GOTTESMAN-KNILL THEOREM

Introductory concepts

➡ The stabiliser formalism and the Gottesman-Knill theorem

- Stabiliser circuits are not universal for quantum computation.

- Nevertheless, the Clifford $+ \, T$ set is universal for quantum computation,

$$T = \mathrm{diag}(1, e^{i\pi/4})\,.$$

Paper review

➡ The extra ingredients

3 different binary classes are considered:

- Stabiliser state inputs (IN(BITS)) vs. More general product states (IN(PROD))
- Adaptivity (ADAPT) vs. Non-adaptivity (NON-ADAPT)
- Single output bit (OUT(1)) vs. Many output bits (OUT(MANY))

Additionally, the classical simulation complexity is considered for both strong and weak notions.

Paper review

➡ The extended classes

	NON-ADAPT	ADAPT
OUT(1)	see tables below	see tables below
OUT(MANY)	see tables below	see tables below

NON-ADAPT, OUT(1)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(1)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

NON-ADAPT, OUT(MANY)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(MANY)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ The extended classes

NON-ADAPT

ADAPT

OUT(1)

NON-ADAPT, OUT(1):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(1):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

GOAL: Determine what is the complexity of classically simulating each of these classes of quantum circuits.

OUT(MANY)

NON-ADAPT, OUT(MANY):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(MANY):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ **The extended classes**

	NON-ADAPT	ADAPT
OUT(1)	see table 1	see table 2
OUT(MANY)	see table 3	see table 4

NON-ADAPT, OUT(1):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(1):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

NON-ADAPT, OUT(MANY):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT, OUT(MANY):

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

*Let $\mathscr{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with computational basis input states and measurements on multiple output qubits. Then, $\mathscr{T}$ can be **classically efficiently simulable** in the strong sense.*

- Input: $\left|\mathbf{x}\right\rangle = \left|0\right\rangle^{\otimes N} = \left|0^N\right\rangle$
- Circuit: C
- Output state: $\left|\psi\right\rangle = C\left|0^N\right\rangle$
- Desired probability: $p = p(\mathbf{y});\ \ \mathbf{y} = 0^M;\ \ M \leq N$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

*Let $\mathscr{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with computational basis input states and measurements on multiple output qubits. Then, $\mathscr{T}$ can be **classically efficiently simulable** in the strong sense.*

- Input: $\left|\mathbf{x}\right\rangle = \left|0\right\rangle^{\otimes N} = \left|0^N\right\rangle$
- Circuit: C
- Output state: $\left|\psi\right\rangle = C\left|0^N\right\rangle$
- Desired probability: $p = p(\mathbf{y});\ \ \mathbf{y} = 0^M;\ \ M \leq N$

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

*Let $\mathscr{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with computational basis input states and measurements on multiple output qubits. Then, $\mathscr{T}$ can be **classically efficiently simulable** in the strong sense.*

- Input: $\left|\mathbf{x}\right\rangle = \left|0\right\rangle^{\otimes N} = \left|0^N\right\rangle$
- Circuit: C
- Output state: $\left|\psi\right\rangle = C\left|0^N\right\rangle$
- Desired probability: $p = p(\mathbf{y});\ \ \mathbf{y} = 0^M;\ \ M \leq N$

$$\boxed{p = \langle\psi|\left(\left|0^M\right\rangle\left\langle 0^M\right|\right)|\psi\rangle}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$p = \langle\psi| \left(\left|0^M\right\rangle \left\langle 0^M\right| \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right) |\psi\rangle$$

$$= \langle\psi| \left(\frac{I+Z}{2}\right)^{\otimes M} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} |\psi\rangle$$

$$= \frac{1}{2^M} \left\langle 0^N\right| C^\dagger \left[\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right) \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C \left|0^N\right\rangle$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$
\begin{aligned}
p &= \langle\psi| \left(|0^M\rangle\langle 0^M| \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right) |\psi\rangle \\
&= \langle\psi| \left(\frac{I+Z}{2} \right)^{\otimes M} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} |\psi\rangle \\
&= \frac{1}{2^M} \langle 0^N| C^\dagger \left[\boxed{\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right)} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C |0^N\rangle
\end{aligned}
$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \langle\psi| \left(\left|0^M\right\rangle \left\langle 0^M\right| \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right) |\psi\rangle$$

$$= \langle\psi| \left(\frac{I+Z}{2} \right)^{\otimes M} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} |\psi\rangle$$

$$= \frac{1}{2^M} \langle 0^N| C^\dagger \left[\boxed{\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right)} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C \left|0^N\right\rangle$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$p = \langle\psi| \left(\left|0^M\right\rangle \left\langle 0^M\right| \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right) |\psi\rangle$$

$$= \langle\psi| \left(\frac{I+Z}{2} \right)^{\otimes M} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \, |\psi\rangle$$

$$= \frac{1}{2^M} \langle 0^N| \, C^\dagger \left[\boxed{\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right)} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C \, |0^N\rangle$$

$$\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right) = \sum_{\mathbf{t} \in \mathbb{Z}_2^M} Z(\mathbf{t})$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \langle 0^N | \, C^\dagger \left[\boxed{\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right)} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C \, | 0^N \rangle$$

$$\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right) = \sum_{\mathbf{t} \in \mathbb{Z}_2^M} Z(\mathbf{t})$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \langle 0^N | \, C^\dagger \left[\boxed{\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right)} \otimes I_{(M+1)} \otimes \ldots \otimes I_{(N)} \right] C \, | 0^N \rangle$$

$$\left(I_{(1)} + Z_{(1)}\right) \otimes \left(I_{(2)} + Z_{(2)}\right) \otimes \ldots \otimes \left(I_{(M)} + Z_{(M)}\right) = \sum_{\mathbf{t} \in \mathbb{Z}_2^M} Z(\mathbf{t})$$

$$Z(\mathbf{t}) \equiv Z_{(1)}^{t_1} \otimes Z_{(2)}^{t_2} \otimes \ldots \otimes Z_{(M)}^{t_M}$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

$$P(\mathbf{t}) = \gamma(\mathbf{t}) \left(X_{(1)}^{a_1(\mathbf{t})} Z_{(1)}^{b_1(\mathbf{t})} \otimes X_{(2)}^{a_2(\mathbf{t})} Z_{(2)}^{b_2(\mathbf{t})} \otimes \ldots \otimes X_{(N)}^{a_N(\mathbf{t})} Z_{(N)}^{b_N(\mathbf{t})} \right)$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \langle 0^N | \, C^\dagger \tilde{Z}(\mathbf{t}) C \, | 0^N \rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

$$P(\mathbf{t}) = \gamma(\mathbf{t}) \left(X_{(1)}^{a_1(\mathbf{t})} Z_{(1)}^{b_1(\mathbf{t})} \otimes X_{(2)}^{a_2(\mathbf{t})} Z_{(2)}^{b_2(\mathbf{t})} \otimes \ldots \otimes X_{(N)}^{a_N(\mathbf{t})} Z_{(N)}^{b_N(\mathbf{t})} \right)$$

$$= \gamma(\mathbf{t}) X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right)$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

$$P(\mathbf{t}) = \gamma(\mathbf{t}) \left(X_{(1)}^{a_1(\mathbf{t})} Z_{(1)}^{b_1(\mathbf{t})} \otimes X_{(2)}^{a_2(\mathbf{t})} Z_{(2)}^{b_2(\mathbf{t})} \otimes \ldots \otimes X_{(N)}^{a_N(\mathbf{t})} Z_{(N)}^{b_N(\mathbf{t})} \right)$$

$$= \gamma(\mathbf{t}) X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right)$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

$$P(\mathbf{t}) = \gamma(\mathbf{t}) \left(X_{(1)}^{a_1(\mathbf{t})} Z_{(1)}^{b_1(\mathbf{t})} \otimes X_{(2)}^{a_2(\mathbf{t})} Z_{(2)}^{b_2(\mathbf{t})} \otimes \ldots \otimes X_{(N)}^{a_N(\mathbf{t})} Z_{(N)}^{b_N(\mathbf{t})} \right)$$

$$= \gamma(\mathbf{t}) X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right)$$

$$\gamma(\mathbf{t}) : \mathbb{Z}_2^M \rightarrow \{\pm 1, \pm i\}$$

Paper review

➡ ***Theorem 4 : STRONG | NON-ADAPT | IN(BITS) | OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$P(\mathbf{t}) = C^\dagger Z(\mathbf{t}) C \Rightarrow P(\mathbf{t})^2 = I$$

$$P(\mathbf{t}) = \gamma(\mathbf{t}) \left(X_{(1)}^{a_1(\mathbf{t})} Z_{(1)}^{b_1(\mathbf{t})} \otimes X_{(2)}^{a_2(\mathbf{t})} Z_{(2)}^{b_2(\mathbf{t})} \otimes \ldots \otimes X_{(N)}^{a_N(\mathbf{t})} Z_{(N)}^{b_N(\mathbf{t})} \right)$$

$$= \gamma(\mathbf{t}) X \left(\boldsymbol{a}(\mathbf{t}) \right) Z \left(\boldsymbol{b}(\mathbf{t}) \right)$$

$$\gamma(\mathbf{t}) : \mathbb{Z}_2^M \rightarrow \{\pm 1, \pm i\} \qquad \boldsymbol{a}, \boldsymbol{b} : \mathbb{Z}_2^M \rightarrow \mathbb{Z}_2^N$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \qquad \rightarrow \quad 2^M \text{ terms}$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \langle 0^N \rvert \, C^\dagger \tilde{Z}(\mathbf{t}) C \, \lvert 0^N \rangle \qquad \rightarrow \; 2^M \text{ terms}$$

$$= \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \gamma(\mathbf{t}) \langle 0^N \rvert \, X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right) \, \lvert 0^N \rangle$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \quad \rightarrow \quad 2^M \text{ terms}$$

$$= \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \gamma(\mathbf{t}) \left\langle 0^N \right| X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right) \left| 0^N \right\rangle$$

$$Z \left| 0 \right\rangle = \left| 0 \right\rangle; \;\; \left\langle 0 \right| X \left| 0 \right\rangle = 0 \Rightarrow X\left(\boldsymbol{a}(\mathbf{t})\right) = I \Rightarrow \boldsymbol{a}(\mathbf{t}) = 0^N$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \qquad \rightarrow \quad 2^M \text{ terms}$$

$$= \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \gamma(\mathbf{t}) \left\langle 0^N \right| X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right) \left| 0^N \right\rangle$$

$$Z\left|0\right\rangle = \left|0\right\rangle; \ \ \left\langle 0\right| X \left|0\right\rangle = 0 \Rightarrow X\left(\boldsymbol{a}(\mathbf{t})\right) = I \Rightarrow \boldsymbol{a}(\mathbf{t}) = 0^N$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \left\langle 0^N \right| C^\dagger \tilde{Z}(\mathbf{t}) C \left| 0^N \right\rangle \qquad \rightarrow \quad 2^M \text{ terms}$$

$$= \frac{1}{2^M} \sum_{\mathbf{t} \in \mathbb{Z}_2^M} \gamma(\mathbf{t}) \left\langle 0^N \right| X\left(\boldsymbol{a}(\mathbf{t})\right) Z\left(\boldsymbol{b}(\mathbf{t})\right) \left| 0^N \right\rangle$$

$$Z \left| 0 \right\rangle = \left| 0 \right\rangle; \;\; \left\langle 0 \right| X \left| 0 \right\rangle = 0 \Rightarrow X\left(\boldsymbol{a}(\mathbf{t})\right) = I \Rightarrow \boldsymbol{a}(\mathbf{t}) = 0^N$$

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} \gamma(\mathbf{t}); \;\; T_0 = \{\mathbf{t} : \boldsymbol{a}(\mathbf{t}) = 0^N\}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

Recalling that $P(\mathbf{t})^2 = I \Rightarrow \gamma(\mathbf{t}) = \pm 1 \equiv (-1)^{u(\mathbf{t})}$

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{u(\mathbf{t})}; \;\; T_0 = \{\mathbf{t} : \boldsymbol{a}(\mathbf{t}) = 0^N\}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

Recalling that $P(\mathbf{t})^2 = I \Rightarrow \gamma(\mathbf{t}) = \pm 1 \equiv (-1)^{u(\mathbf{t})}$

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{u(\mathbf{t})}; \;\; T_0 = \{\mathbf{t} : \boldsymbol{a}(\mathbf{t}) = 0^N\}$$

Prove that:

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

Recalling that $P(\mathbf{t})^2 = I \Rightarrow \gamma(\mathbf{t}) = \pm 1 \equiv (-1)^{u(\mathbf{t})}$

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{u(\mathbf{t})}; \;\; T_0 = \{\mathbf{t} : \boldsymbol{a}(\mathbf{t}) = 0^N\}$$

Prove that:

(i) T_0 can be classically determined in polynomial time;

Paper review

➡ ***Theorem 4 : STRONG | NON-ADAPT | IN(BITS) | OUT(MANY)***

Recalling that $P(\mathbf{t})^2 = I \Rightarrow \gamma(\mathbf{t}) = \pm 1 \equiv (-1)^{u(\mathbf{t})}$

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{u(\mathbf{t})}; \; T_0 = \{\mathbf{t} : \boldsymbol{a}(\mathbf{t}) = 0^N\}$$

Prove that:

(i) T_0 can be classically determined in polynomial time;

(ii) The sum can be classically efficiently computed.

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(i) T_0 can be classically determined in polynomial time:

Define a basis of $\mathbb{Z}_2^M$, $\{\boldsymbol{e}_i\,,\ i = 1,\ldots,M\}$: $\boldsymbol{e}_i = 0_1 0_2 \ldots 1_i \ldots 0_M$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(i) T_0 can be classically determined in polynomial time:

Define a basis of $\mathbb{Z}_2^M$, $\{\boldsymbol{e}_i\,,\ i = 1,...,M\}\ :\ \boldsymbol{e}_i = 0_1 0_2 \ldots 1_i \ldots 0_M$

Then, any bit string can be written as: $\mathbf{t} = \sum_{k=1}^{M} t_k \boldsymbol{e}_k$

Paper review

➡ *Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)*

(i) T_0 can be classically determined in polynomial time:

Define a basis of $\mathbb{Z}_2^M$, $\{\boldsymbol{e}_i\,,\ i = 1,\ldots,M\}\ :\ \boldsymbol{e}_i = 0_1 0_2 \ldots 1_i \ldots 0_M$

Then, any bit string can be written as: $\mathbf{t} = \sum_{k=1}^{M} t_k \boldsymbol{e}_k$

Paper review

➡ *Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)*

(i) T_0 can be classically determined in polynomial time:

Define a basis of $\mathbb{Z}_2^M$, $\{\boldsymbol{e}_i\, ,\ i = 1,...,M\} : \boldsymbol{e}_i = 0_1 0_2 \dots 1_i \dots 0_M$

Then, any bit string can be written as: $\mathbf{t} = \sum_{k=1}^{M} t_k \boldsymbol{e}_k$

And it is also possible to write: $\boldsymbol{a}(\mathbf{t}) = \sum_{k=1}^{M} t_k \boldsymbol{a}(\boldsymbol{e}_k)$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(i) T_0 can be classically determined in polynomial time (cont.):

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(i) T_0 can be classically determined in polynomial time (cont.):

The labels $\boldsymbol{a}(\boldsymbol{e}_i)$ can be efficiently computed from the Clifford conjugation rules.

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(i) T_0 can be classically determined in polynomial time (cont.):

The labels $\boldsymbol{a}(\boldsymbol{e}_i)$ can be efficiently computed from the Clifford conjugation rules.

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(i) T_0 can be classically determined in polynomial time (cont.):

The labels $\boldsymbol{a}(\boldsymbol{e}_i)$ can be efficiently computed from the Clifford conjugation rules.

They can be used to construct the columns of an $N \times M$ matrix A such that:
$T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\} \equiv \ker(A)\,.$

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(i) T_0 can be classically determined in polynomial time (cont.):

The labels $\boldsymbol{a}(\boldsymbol{e}_i)$ can be efficiently computed from the Clifford conjugation rules.

They can be used to construct the columns of an $N \times M$ matrix A such that:
$T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\} \equiv \ker(A)$.

Denote the basis of the kernel of A as $\{\boldsymbol{c}_i,\ i = 1,\ldots,L \leq M\}$.

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(i) T_0 can be classically determined in polynomial time (cont.):

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(i) T_0 can be classically determined in polynomial time (cont.):

There are classical algorithms which allow the efficient determination of the basis of the kernel of a matrix, so the first statement is proved.

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed:

Note that $\mathbf{t} \in T_0 \text{ iff } A\mathbf{t} = 0^N \Leftrightarrow \mathbf{t} \in T_0 \text{ iff } \mathbf{t} = \sum_{k=1}^{L} s_k \boldsymbol{c}_k \,.$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed:

Note that $\mathbf{t} \in T_0$ iff $A\mathbf{t} = 0^N \Leftrightarrow \mathbf{t} \in T_0$ iff $\mathbf{t} = \sum_{k=1}^{L} s_k \boldsymbol{c}_k$.

Therefore, $u(\mathbf{t}) = u\left(\sum_{k=1}^{L} s_k \boldsymbol{c}_k\right) = \sum_{k=1}^{L} s_k u\left(\boldsymbol{c}_k\right)$.

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed:

Note that $\mathbf{t} \in T_0$ iff $A\mathbf{t} = 0^N \Leftrightarrow \mathbf{t} \in T_0$ iff $\mathbf{t} = \sum_{k=1}^{L} s_k \boldsymbol{c}_k$.

Therefore, $u(\mathbf{t}) = u\left(\sum_{k=1}^{L} s_k \boldsymbol{c}_k\right) = \sum_{k=1}^{L} s_k u\left(\boldsymbol{c}_k\right)$.

Let $u(\boldsymbol{c}_k) = q_k \to u(\mathbf{t}) = \boldsymbol{s} \cdot \boldsymbol{q}$

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{\boldsymbol{u}(\mathbf{t})} = \frac{1}{2^M} \sum_{\boldsymbol{s} \in \mathbb{Z}_2^L} (-1)^{\boldsymbol{s} \cdot \boldsymbol{q}}; \;\; T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{u(\mathbf{t})} = \frac{1}{2^M} \sum_{s \in \mathbb{Z}_2^L} (-1)^{s \cdot q}; \;\; T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{\boldsymbol{u}(\mathbf{t})} = \frac{1}{2^M} \sum_{s \in \mathbb{Z}_2^L} (-1)^{\boldsymbol{s} \cdot \boldsymbol{q}}; \; T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

$$p = \begin{cases} (1/2)^{M-L}, & \text{if } \boldsymbol{q} = 0^L \\ 0, & \text{if } \boldsymbol{q} \neq 0^L \end{cases}$$

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{\boldsymbol{u}(\mathbf{t})} = \frac{1}{2^M} \sum_{s \in \mathbb{Z}_2^L} (-1)^{\boldsymbol{s} \cdot \boldsymbol{q}}; \;\; T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

$$p = \begin{cases} (1/2)^{M-L}, & \text{if } \boldsymbol{q} = 0^L \\ 0, & \text{if } \boldsymbol{q} \neq 0^L \end{cases}$$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{\boldsymbol{u}(\mathbf{t})} = \frac{1}{2^M} \sum_{s \in \mathbb{Z}_2^L} (-1)^{\boldsymbol{s} \cdot \boldsymbol{q}}; \ T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

$$\boxed{p = \begin{cases} (1/2)^{M-L}, & \text{if } \boldsymbol{q} = 0^L \\ 0, & \text{if } \boldsymbol{q} \neq 0^L \end{cases}}$$

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

(ii) The sum can be classically efficiently computed (cont.):

Returning to the sum we have:

$$p = \frac{1}{2^M} \sum_{\mathbf{t} \in T_0} (-1)^{\boldsymbol{u}(\mathbf{t})} = \frac{1}{2^M} \sum_{s \in \mathbb{Z}_2^L} (-1)^{s \cdot \boldsymbol{q}}; \; T_0 = \{\mathbf{t} : A\mathbf{t} = 0^N\}$$

$$p = \begin{cases} (1/2)^{M-L}, & \text{if } \boldsymbol{q} = 0^L \\ 0, & \text{if } \boldsymbol{q} \neq 0^L \end{cases}$$

Strong simulation of this family of circuits can be carried out efficiently

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

Summary | Procedure for the efficient strong classical simulation:

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

Summary | Procedure for the efficient strong classical simulation:

1. determine the $M\,\boldsymbol{a}\left(\boldsymbol{e}_k\right)$ labels efficiently from the Clifford update rules;

Paper review

➡ *Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)*

Summary | Procedure for the efficient strong classical simulation:

1. determine the M $\boldsymbol{a}\left(\boldsymbol{e}_k\right)$ labels efficiently from the Clifford update rules;

2. construct the $N \times M$ matrix A;

Paper review

➡ *Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)*

Summary | Procedure for the efficient strong classical simulation:

1. determine the $M\ \boldsymbol{a}\left(\boldsymbol{e}_k\right)$ labels efficiently from the Clifford update rules;

2. construct the $N \times M$ matrix A;

3. determine the $\ker(A)$ and its basis $\{\boldsymbol{c}_k\}$ (classically efficient);

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

Summary | Procedure for the efficient strong classical simulation:

1. determine the $M\ \boldsymbol{a}\left(\boldsymbol{e}_k\right)$ labels efficiently from the Clifford update rules;

2. construct the $N \times M$ matrix A;

3. determine the $\ker(A)$ and its basis $\{\boldsymbol{c}_k\}$ (classically efficient);

4. for each $\boldsymbol{c}_k$ compute: $q_k = u(\boldsymbol{c}_k)$ using the Clifford update rules;

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

Summary | Procedure for the efficient strong classical simulation:

1. determine the $M\ \boldsymbol{a}\left(\boldsymbol{e}_k\right)$ labels efficiently from the Clifford update rules;

2. construct the $N \times M$ matrix A;

3. determine the $\ker(A)$ and its basis $\{\boldsymbol{c}_k\}$ (classically efficient);

4. for each $\boldsymbol{c}_k$ compute: $q_k = u(\boldsymbol{c}_k)$ using the Clifford update rules;

5. If $\boldsymbol{q} = 0^L, p = (1/2)^{M-L},$ while $\boldsymbol{q} \neq 0^L \Rightarrow p = 0.$

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

NON-ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

NON-ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)		*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ ***Theorem 4 : STRONG|NON-ADAPT|IN(BITS)|OUT(MANY)***

NON-ADAPT — **OUT(1)**

	WEAK	STRONG
IN(BITS)		*Clas. Effic.*
IN(PROD)		

ADAPT — **OUT(1)**

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

NON-ADAPT — **OUT(MANY)**

	WEAK	STRONG
IN(BITS)		*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT — **OUT(MANY)**

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ ***Theorem 4 : STRONG| NON-ADAPT| IN(BITS)| OUT(MANY)***

OUT(1), NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)		

OUT(1), ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

OUT(MANY), NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY), ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ *Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)*

*Let $\mathcal{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with general product state input and measurement of a single output qubit. Then, $\mathcal{T}$ can be **classically efficiently simulable** in the strong sense.*

- Input: $\left|\psi_0\right\rangle = \left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \ldots \left|\alpha_N\right\rangle$
- Circuit: C
- Output state: $\left|\psi_f\right\rangle = C\left|\psi_0\right\rangle = C\left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \ldots \left|\alpha_N\right\rangle$
- Output: $b = 0$ or $b = 1$, with probabilities p_0 and p_1 .

Paper review

➡ *Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)*

*Let $\mathcal{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with general product state input and measurement of a single output qubit. Then, $\mathcal{T}$ can be **classically efficiently simulable** in the strong sense.*

- Input: $\left|\psi_0\right\rangle = \left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \ldots \left|\alpha_N\right\rangle$
- Circuit: C
- Output state: $\left|\psi_f\right\rangle = C\left|\psi_0\right\rangle = C\left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \ldots \left|\alpha_N\right\rangle$
- Output: $b = 0$ or $b = 1$, with probabilities p_0 and p_1 .

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

Let $\mathcal{T}$ be a set of computational tasks defined by non-adaptive Clifford circuits, with general product state input and measurement of a single output qubit. Then, $\mathcal{T}$ can be ***classically efficiently simulable*** *in the strong sense.*

- Input: $\left|\psi_0\right\rangle = \left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \dots \left|\alpha_N\right\rangle$
- Circuit: C
- Output state: $\left|\psi_f\right\rangle = C\left|\psi_0\right\rangle = C\left|\alpha_1\right\rangle \left|\alpha_2\right\rangle \dots \left|\alpha_N\right\rangle$
- Output: $b = 0$ or $b = 1$, with probabilities p_0 and p_1 .

$$p_0 = \left\langle\psi_f\right| \left(\left|0\right\rangle\left\langle 0\right|\right) \left|\psi_f\right\rangle$$
$$p_1 = \left\langle\psi_f\right| \left(\left|1\right\rangle\left\langle 1\right|\right) \left|\psi_f\right\rangle$$

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

The two probabilities can be written as:

$$p_0 = \langle \psi_0 | \, C^\dagger \left(\frac{I+Z}{2} \otimes I \otimes \ldots \otimes I \right) C \, | \psi_0 \rangle$$

$$p_1 = \langle \psi_0 | \, C^\dagger \left(\frac{I-Z}{2} \otimes I \otimes \ldots \otimes I \right) C \, | \psi_0 \rangle$$

And therefore the difference between them reads:

$$p_0 - p_1 = \langle \psi_0 | \, C^\dagger \, (Z \otimes \ldots \otimes I) \, C \, | \psi_0 \rangle$$

Paper review

➡ ***Theorem 1 : STRONG | NON-ADAPT | IN(PROD) | OUT(1)***

The two probabilities can be written as:

$$p_0 = \langle \psi_0 | \, C^\dagger \left(\frac{I+Z}{2} \otimes I \otimes \ldots \otimes I \right) C \, | \psi_0 \rangle$$

$$p_1 = \langle \psi_0 | \, C^\dagger \left(\frac{I-Z}{2} \otimes I \otimes \ldots \otimes I \right) C \, | \psi_0 \rangle$$

And therefore the difference between them reads:

$$p_0 - p_1 = \langle \psi_0 | \boxed{C^\dagger (Z \otimes \ldots \otimes I) \, C} | \psi_0 \rangle$$

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

The two probabilities can be written as:

$$p_0 = \langle\psi_0|\, C^\dagger \left(\frac{I+Z}{2} \otimes I \otimes \ldots \otimes I\right) C\,|\psi_0\rangle$$

$$p_1 = \langle\psi_0|\, C^\dagger \left(\frac{I-Z}{2} \otimes I \otimes \ldots \otimes I\right) C\,|\psi_0\rangle$$

And therefore the difference between them reads:

$$p_0 - p_1 = \langle\psi_0|\,\boxed{C^\dagger (Z \otimes \ldots \otimes I)\, C}\,|\psi_0\rangle$$

Pauli operator

Paper review

➡ ***Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)***

$C^\dagger (Z \otimes \ldots \otimes I)\, C = \pm\, P_{(1)} \otimes P_{(2)} \otimes \ldots \otimes P_{(N)}$ (efficiently determined from the Clifford update rules)

Paper review

➡ ***Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)***

$C^\dagger (Z \otimes \ldots \otimes I)\, C = \pm P_{(1)} \otimes P_{(2)} \otimes \ldots \otimes P_{(N)}$ (efficiently determined from the Clifford update rules)

Therefore the difference between the two probabilities is simply:

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

$C^{\dagger}\,(Z \otimes \ldots \otimes I)\, C = \pm\, P_{(1)} \otimes P_{(2)} \otimes \ldots \otimes P_{(N)}$ (efficiently determined from the Clifford update rules)

Therefore the difference between the two probabilities is simply:

$$p_0 - p_1 = \pm \Pi_{k=1}^{N} \left\langle \alpha_k \right| P_{(k)} \left| \alpha_k \right\rangle .$$

Paper review

➡ ***Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)***

$C^{\dagger}\,(Z \otimes \ldots \otimes I)\,C = \pm P_{(1)} \otimes P_{(2)} \otimes \ldots \otimes P_{(N)}$ (efficiently determined from the Clifford update rules)

Therefore the difference between the two probabilities is simply:

$$p_0 - p_1 = \pm \Pi_{k=1}^{N} \left\langle \alpha_k \middle| P_{(k)} \middle| \alpha_k \right\rangle .$$

Paper review

➡ ***Theorem 1 : STRONG|NON-ADAPT|IN(PROD)|OUT(1)***

$C^{\dagger}\,(Z \otimes \ldots \otimes I)\,C = \pm\, P_{(1)} \otimes P_{(2)} \otimes \ldots \otimes P_{(N)}$ (efficiently determined from the Clifford update rules)

Therefore the difference between the two probabilities is simply:

$$p_0 - p_1 = \pm\, \Pi_{k=1}^{N} \left\langle \alpha_k \middle| \, P_{(k)} \, \middle| \alpha_k \right\rangle .$$

We need only calculate N expectation values of 2×2 Pauli matrices, which can be done classically in $poly(N)$ time.

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

OUT(1) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)		*Clas. Effic. (Theorem 1)*

OUT(1) — ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

OUT(MANY) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY) — ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ ***Theorem 1 : STRONG| NON-ADAPT| IN(PROD)| OUT(1)***

OUT(1)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

OUT(MANY)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ *Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)*

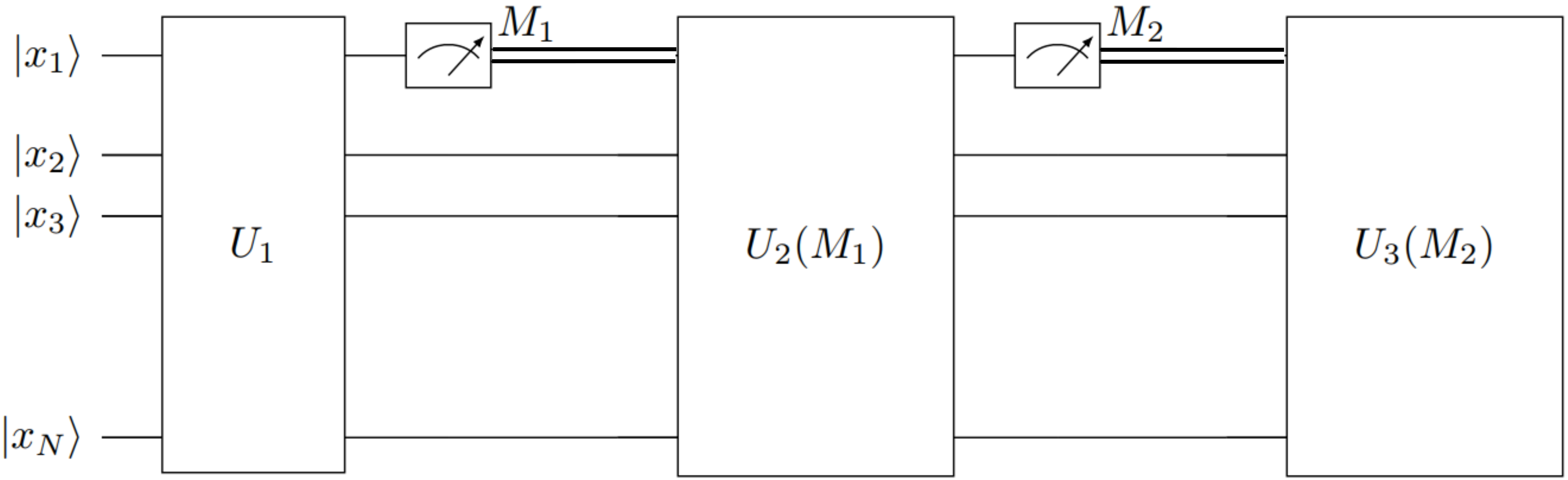
*Let $\mathscr{T}$ be a set of computational tasks defined by **adaptive** Clifford circuits, with computational basis input states and measurements on multiple output qubits. Then, $\mathscr{T}$ can be **classically efficiently simulable** in the weak sense.*

- Input: $|\mathbf{x}\rangle = \left|x_1x_2\ldots x_N\right\rangle$
- Circuit: C
- K intermediate measurements + M output measurements
- Output distribution: $p = p(\mathbf{y}) = p(y_1, y_2, \ldots, y_{K+M})$

Paper review

➡ *Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)*

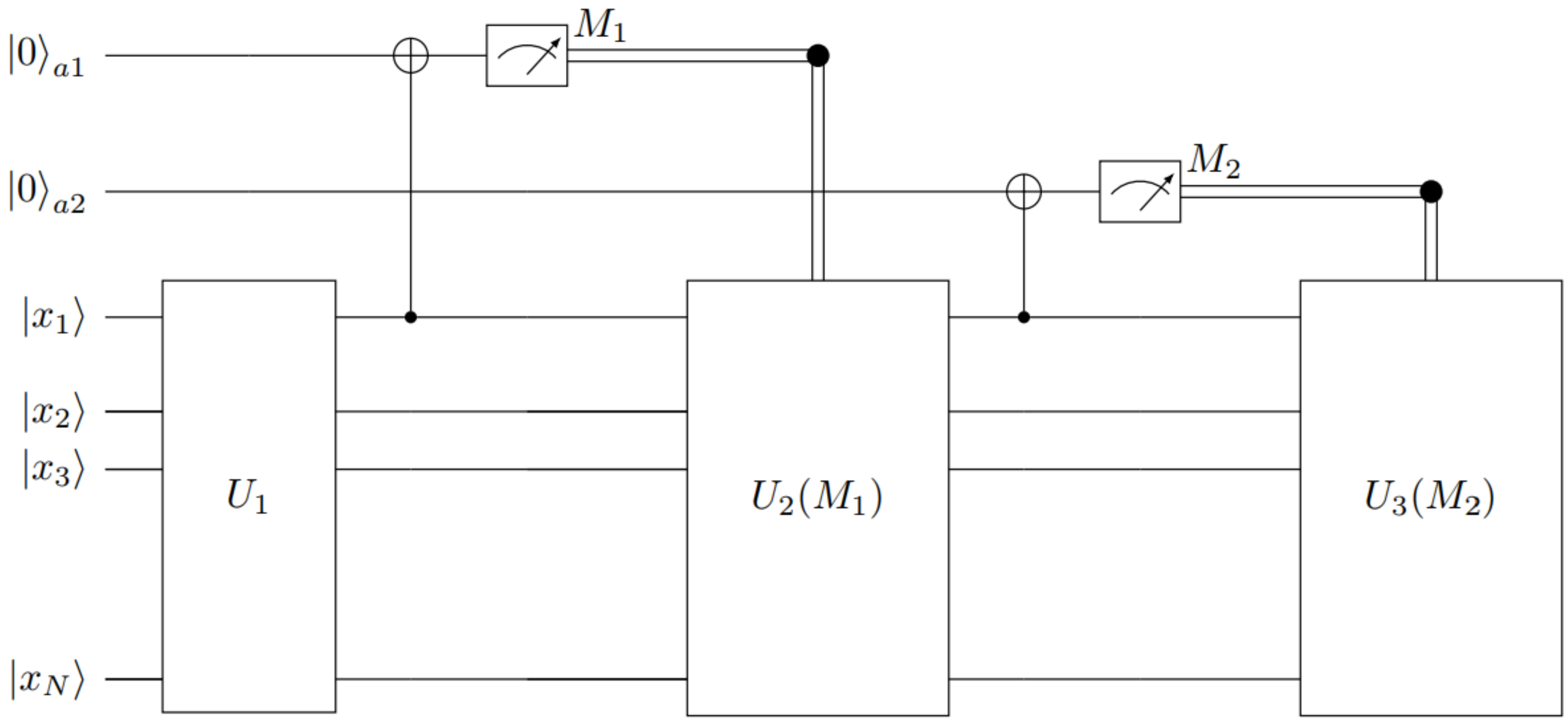
- Consider a circuit C such as:



Paper review

➡ *Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)*

- Consider a circuit C' on $N + K$ qubits so that:



Paper review

➡ ***Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)***

- C and C' are equivalent and for C' we have:

(i) input state $\left|0_1 \ldots 0_K x_{K+1} \ldots x_{K+N}\right\rangle = \left|0_1 \ldots 0_K\right\rangle |\mathbf{x}\rangle$;

(ii) output measurements are carried out on qubits $K+1$ to $K+M$;

(iii) intermediate measurements are carried out on the first K qubits, and those are not used thereafter.

Paper review

➡ *Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)*

- A full run of C' samples an associated probability distribution $p(y_1 \dots y_K y_{K+1} \dots y_{K+M})$.

- Suppose that all intermediate measurements have been carried out.

- Then, the sequence $y_1 \dots y_K$ is fixed and the circuit C' becomes non-adaptive.

Paper review

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

	NON-ADAPT	ADAPT
OUT(1)	see below	see below
OUT(MANY)	see below	see below

OUT(1), NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(1), ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

OUT(MANY), NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY), ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

Paper review

➡ ***Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)***

- Then we can efficiently compute the marginal probabilities $p(y_1 \ldots y_K)$ and $p(y_1 \ldots y_K y_{K+1} \ldots y_{K+N})$.

- This means that we know the probability of occurrence of each possible non-adaptive circuit C'; and for each of those we know the probability of each string.

- Therefore, we can sample from this distribution and weakly simulate the adaptive circuit C' and, thus, C.

Paper review

➡ ***Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)***

OUT(1) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(1) — ADAPT

	WEAK	STRONG
IN(BITS)		
IN(PROD)		

OUT(MANY) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY) — ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)		

Paper review

➡ ***Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)***

	NON-ADAPT	ADAPT
OUT(1)	see tables below	see tables below
OUT(MANY)	see tables below	see tables below

OUT(1) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(1) — ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	
IN(PROD)		

OUT(MANY) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY) — ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)		

Paper review

➡ Remarks on the Gottesman-Knill theorem

- **GOTTESMAN-KNILL THEOREM (GK):** (version 1)

  *Any quantum computation carried out on a (potentially **adaptive**) stabiliser circuit can be perfectly **weakly** simulated in polynomial time on a probabilistic classical computer.*

[1] D. Gottesman, in *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics* (1998) pp. 32–43, arXiv:quant-ph/9807006v1.

Paper review

➡ Remarks on the Gottesman-Knill theorem

- **GOTTESMAN-KNILL THEOREM (GK):** (version 2)

  *For any (**non-adaptive**) stabiliser circuit with a single output qubit, the probability p that the output qubit is 1, can be efficiently classically computed.*

[2] S. Aaronson and D. Gottesman, Phys. Rev. A **70**, 052328 (2004), arXiv:quant-ph/0406196v5.

Paper review

➡ ***Theorem 5: WEAK|ADAPT|IN(BITS)|OUT(MANY)***

NON-ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	
IN(PROD)		

NON-ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)		

Paper review

➡ ***Theorem 3: WEAK|ADAPT|IN(PROD)|OUT(1)***

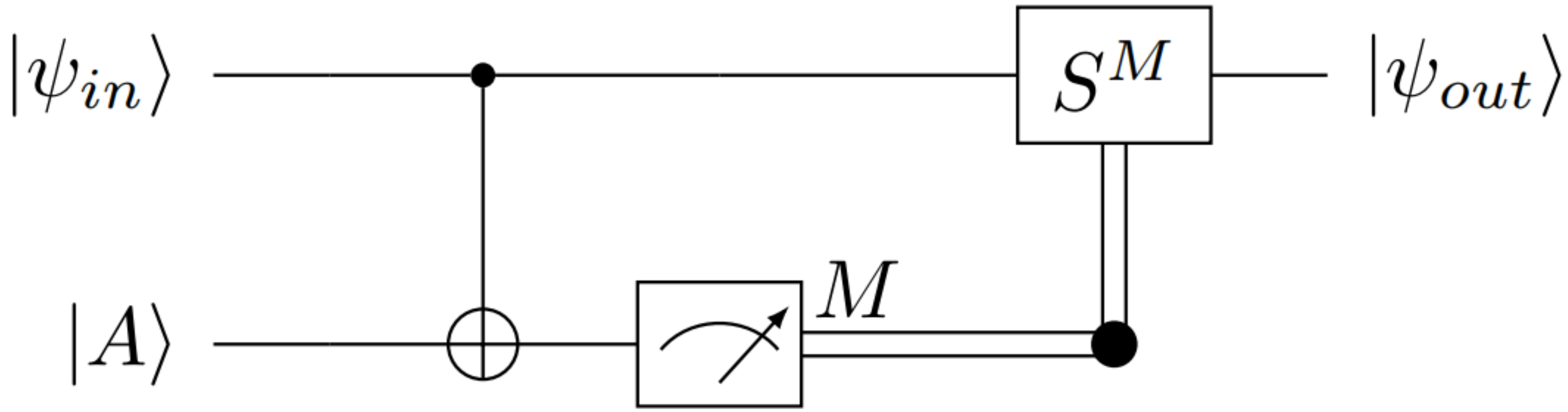
*Let $\mathscr{T}$ be a set of computational tasks defined by **adaptive** Clifford circuits with general product state inputs and output measurement on a single qubit. Then, the weak classical simulation of $\mathscr{T}$ is **QC**-hard.*

- **QC**-hard means that universal quantum computation is possible.

- To prove this it suffices to show that the resources available allow to implement the T gate: $T = \mathrm{diag}\left(1, e^{i\pi/4}\right)$.

Paper review

➡ ***Theorem 3: WEAK| ADAPT| IN(PROD)| OUT(1)***



$$|A\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\pi/4}|1\rangle\right)$$

$$|\psi_{out}\rangle = T|\psi_{in}\rangle$$

Paper review

➡ ***Theorem 3: WEAK|ADAPT|IN(PROD)|OUT(1)***

NON-ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	
IN(PROD)	*Univ. QC (Theorem 3)*	

NON-ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)		

Paper review

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

➡ ***Theorem 3: WEAK|ADAPT|IN(PROD)|OUT(1)***

OUT(1)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	
IN(PROD)	*Univ. QC (Theorem 3)*	

OUT(MANY)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)	*Univ. QC*	

Paper review

➡ ***Theorem 3: WEAK|ADAPT|IN(PROD)|OUT(1)***

OUT(1) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(1) — ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	
IN(PROD)	*Univ. QC (Theorem 3)*	

OUT(MANY) — NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

OUT(MANY) — ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)	*Univ. QC*	

Paper review

➡ ***Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)***

*Consider a set of computational tasks $\mathscr{T}$ defined by **adaptive** Clifford circuits such that input states are computational basis states and only a single output measurement is performed. Then, strong simulation of tasks in $\mathscr{T}$ is #$\mathbf{P}$-hard.*

- The available ingredients can be used to realize the Toffoli gate.

$$TOFF\,|a\rangle\,|b\rangle\,|c\rangle = |a\rangle\,|b\rangle\,|c \oplus (ab)\rangle$$

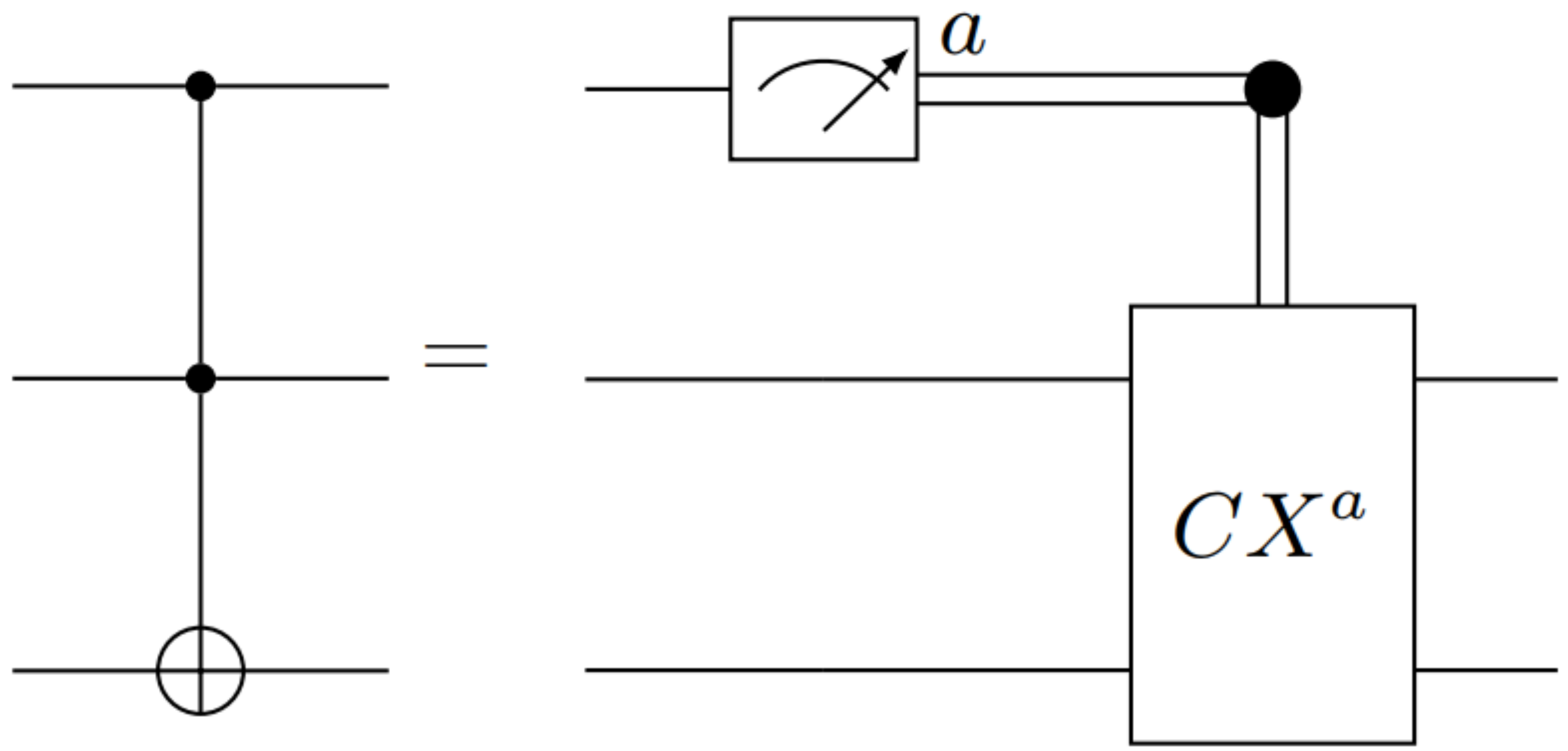
$$a = 0 \Rightarrow TOFF\,|0\rangle\,|b\rangle\,|c\rangle = |0\rangle\,|b\rangle\,|c\rangle \equiv |0\rangle\, I\big(|b\rangle\,|c\rangle\big)$$

$$a = 1 \Rightarrow TOFF\,|1\rangle\,|b\rangle\,|c\rangle = |1\rangle\,|b\rangle\,|c \oplus b\rangle \equiv |1\rangle\, CX\big(|b\rangle\,|c\rangle\big)$$

Paper review

➡ *Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)*

- If the i-th line is promised to be in a computational basis state we can implement the Toffoli gate as;



- This sort of implementation does not allow the application of Toffoli gates coherently on general quantum states, because the adaptation requires a measurement on the i-th line.

Paper review

➡ ***Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)***

- The Toffoli gate can perform universal classical computation.

- Therefore, the defined family of circuits can perform universal classical computation. ⇒ They can compute any Boolean function with an N bit input and a single bit output: $f(\boldsymbol{x}) \in \mathbb{Z}_2$, $\boldsymbol{x} \in \mathbb{Z}_2^N$.

Paper review

➡ ***Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)***

- Procedure to implement a circuit $C \in \mathcal{T}$ on $N+1$ qubits:

  1. Every qubit is initialised in $|0\rangle$;

  2. First N qubits are transformed by a Hadamard gate and then measured generating a random bit-string $\boldsymbol{x} \equiv x_1 x_2 \ldots x_N$;

  3. Perform the following mapping $U \in \mathcal{T} : U\,|\boldsymbol{x}\rangle\,|0\rangle = |\boldsymbol{x}\rangle\,|f(\boldsymbol{x})\rangle$;

  4. Measure the last qubit, registering the value of the function.

Paper review

➡ ***Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)***

$$p(1) = \frac{\#f}{2^N}$$

- If it is possible to determine the probability $p = p(1)$, then it is possible to know $\#f$.

- If it were possible to determine $p(1)$ then it would be possible to count the number of solutions to an $\mathbf{NP}$-hard satisfiability problem, i.e., it would be possible to solve a $\#\boldsymbol{P}$-hard problem.

Paper review

➡ ***Theorem 2: STRONG| ADAPT| IN(BITS)| OUT(1)***

NON-ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT — OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	#**P**-hard *(Theorem 2)*
IN(PROD)	*Univ. QC (Theorem 3)*	

NON-ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT — OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	
IN(PROD)	*Univ. QC*	

Paper review

➡ ***Theorem 2: STRONG|ADAPT|IN(BITS)|OUT(1)***

OUT(1)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	#**P**-hard *(Theorem 2)*
IN(PROD)	*Univ. QC (Theorem 3)*	#**P**-hard

OUT(MANY)

NON-ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		

ADAPT

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	#**P**-hard
IN(PROD)	*Univ. QC*	#**P**-hard

Paper review

➡ ***Theorem 6: STRONG/ NON-ADAPT|IN(PROD)|OUT(MANY)***

Consider a set of computational tasks $\mathscr{T}$ defined by non-adaptive Clifford circuits, with any general product state input and multiple bit output. Then, strong simulation of $\mathscr{T}$ is $\#\mathbf{P}$-hard.

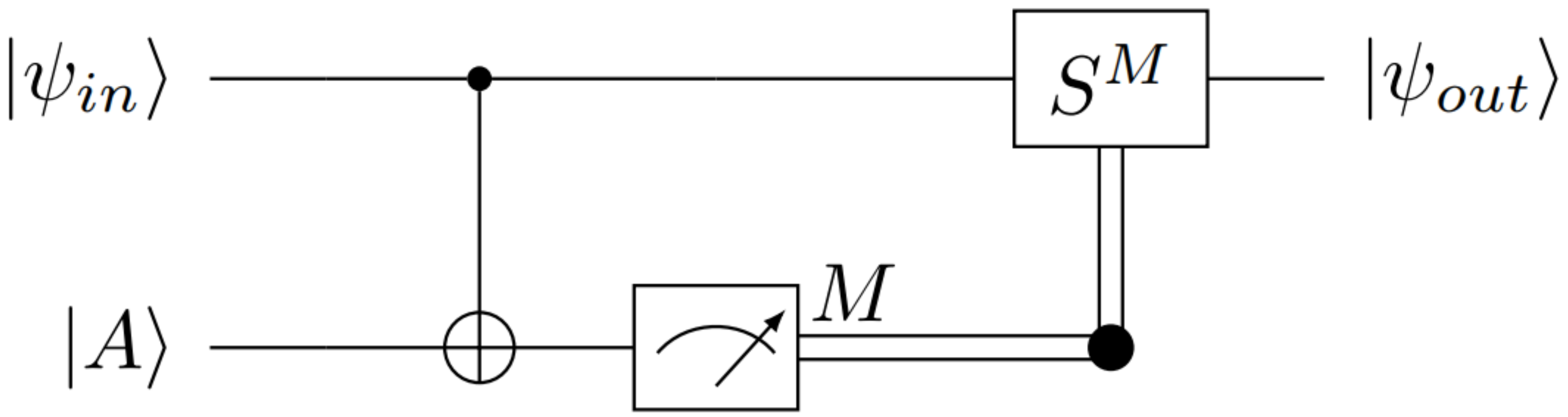
- Consider a universal quantum circuit C, which has K T gates.

- We can turn this into a Clifford circuit C' on $N+K$ qubits, replacing each T gate in a line i by CX_{ia}, a an ancillary magic state qubit.

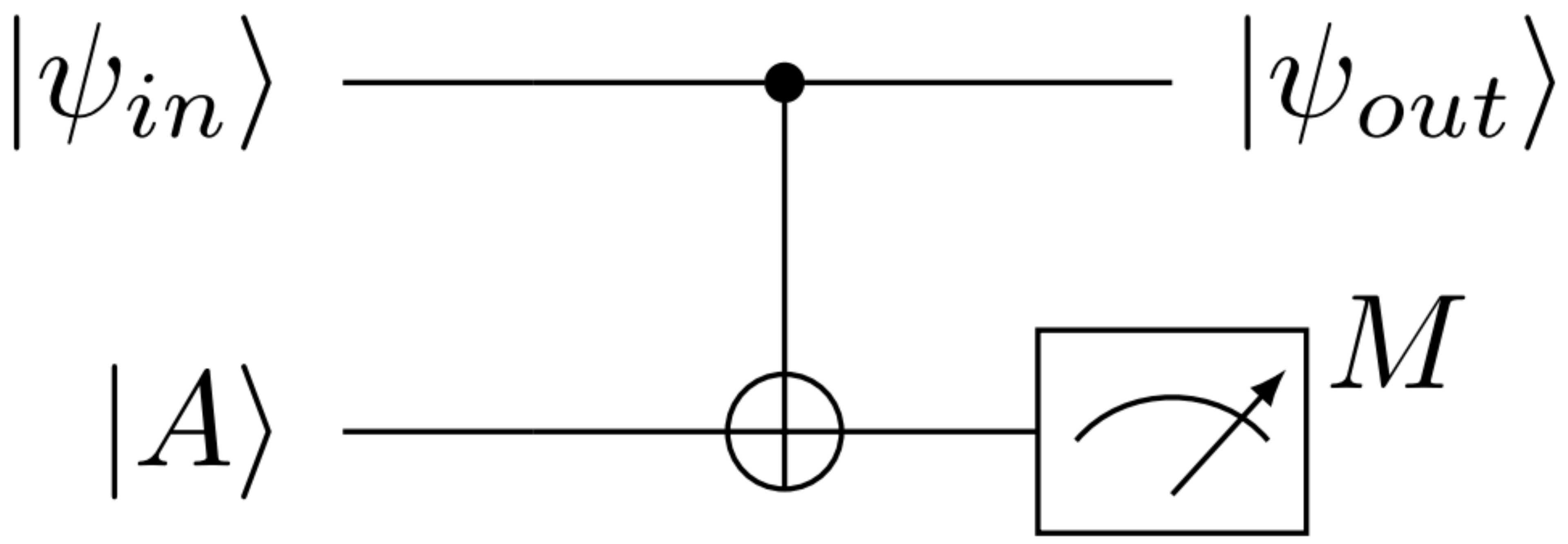
Paper review

➡ *Theorem 6: STRONG/ NON-ADAPT| IN(PROD)| OUT(MANY)*

- Recall the T gadget:



- But now we implement instead:



Paper review

➡ ***Theorem 6: STRONG/ NON-ADAPT| IN(PROD)| OUT(MANY)***

- C and C' only coincide if all K intermediate measurements yield 0 in which case we can write:

$$p_C(\boldsymbol{y}) = p_{C'}(\boldsymbol{y}\,|\,0_1 \ldots 0_K) = \frac{p_{C'}(\boldsymbol{y}0_1 \ldots 0_K)}{p_{C'}(0_1 \ldots 0_K)}\,.$$

- $p_C(\boldsymbol{y})$ could be used to encode $\#f$ of any Boolean function, and solve $\#\boldsymbol{P}$-hard problems.

Paper review

➡ *Theorem 6*

NON-ADAPT

OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)		#**P**-hard *(Theorem 6)*

ADAPT

OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	#**P**-hard *(Theorem 2)*
IN(PROD)	*Univ. QC (Theorem 3)*	#**P**-hard

OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	#**P**-hard
IN(PROD)	*Univ. QC*	#**P**-hard

Paper review

➡ ***Theorem 7: WEAK/ NON-ADAPT|IN(PROD)|OUT(MANY)***

Let $\mathscr{T}$ be the set of computational tasks defined (as in theorem 6) by non-adaptive Clifford circuits, general product state inputs and multiple bit outputs. If $\mathscr{T}$ could be weakly efficiently classically simulated, then the polynomial hierarchy $\mathbf{PH}$ *would collapse to its third level.*

Paper review

➡ ***Theorem 7: WEAK/ NON-ADAPT| IN(PROD)| OUT(MANY)***

NON-ADAPT

OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic.*
IN(PROD)	*Clas. Effic.*	*Clas. Effic. (Theorem 1)*

OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	*Clas. Effic. (Theorem 4)*
IN(PROD)	If Clas. Effic. then PH collapse *(Theorem 7)*	**#P**-hard *(Theorem 6)*

A more recent result shows that under certain circumstances this can actually be classically efficient!

ADAPT

OUT(1)

	WEAK	STRONG
IN(BITS)	*Clas. Effic.*	**#P**-hard *(Theorem 2)*
IN(PROD)	*Univ. QC (Theorem 3)*	**#P**-hard

OUT(MANY)

	WEAK	STRONG
IN(BITS)	*Clas. Effic. (Theorem 5)*	**#P**-hard
IN(PROD)	*Univ. QC*	**#P**-hard

This means **BQP**-complete! Their designation in the paper is a bit unfortunate!

Thank you for your attention!

Introductory concepts

➡ The Clifford group and stabiliser circuits

- Action of the generators of the Clifford group on the Pauli group generators:

$$H_a X_a H_a^\dagger = Z_a; \;\; H_a Z_a H_a^\dagger = X_a; \; \rightarrow \; \text{swap } x_{ia} \text{ and } z_{ia}\,;\, s_i' = s_i \oplus x_{ia} z_{ia}$$

$$S_a X_a S_a^\dagger = Y_a; \;\; S_a Z_a S_a^\dagger = Z_a; \quad \rightarrow \; x_{ia}' = x_{ia}\,;\, z_{ia}' = z_{ia} \oplus x_{ia}\,;\, s_i' = s_i \oplus x_{ia} z_{ia}$$

Introductory concepts

➡ The Clifford group and stabiliser circuits

- Action of the generators of the Clifford group on the Pauli group generators:

$$CX_{ab}(X_{(a)} \otimes I_{(b)})CX^{\dagger}_{ab} = (X_{(a)} \otimes X_{(b)});\ \ CX_{ab}(I_{(a)} \otimes X_{(b)})CX^{\dagger}_{ab} = (I_{(a)} \otimes X_{(b)});$$

$$CX_{ab}(Z_{(a)} \otimes I_{(b)})CX^{\dagger}_{ab} = (Z_{(a)} \otimes I_{(b)});\ \ CX_{ab}(I_{(a)} \otimes Z_{(b)})CX^{\dagger}_{ab} = (Z_{(a)} \otimes Z_{(b)});$$

$$\rightarrow x'_{ia} = x_{ia};\ x'_{ib} = x_{ia} \oplus x_{ib};\ z'_{ia} = z_{ia} \oplus z_{ib};\ z'_{ib} = z_{ib}\,;$$

$$s'_i = s_i \oplus x_{ia} z_{ib} \left(x_{ib} z_{ia} \oplus 1 \right)$$

Paper review

➡ ***Theorem 7: WEAK/ NON-ADAPT|IN(PROD)|OUT(MANY)***

Let $\mathscr{T}$ be the set of computational tasks defined (as in theorem 6) by non-adaptive Clifford circuits, general product state inputs and multiple bit outputs. If $\mathscr{T}$ could be weakly efficiently classically simulated, then the polynomial hierarchy $\mathbf{PH}$ *would collapse to its third level.*

- Again consider a universal quantum circuit C, with K T gates.

- To implement each T gate we use the same gadget as in the previous theorem, post-selecting the value 0 for all of the ancillas.

Paper review

➡ ***Theorem 7: WEAK/ NON-ADAPT|IN(PROD)|OUT(MANY)***

- $\mathscr{T}$ + post-selection **contains** universal quantum computation with post-selection.
- $\text{post}\mathbf{BQP} = \mathbf{PP}$
- Therefore, $\text{post}\mathscr{T}$ contains $\mathbf{PP}$.

Paper review

➡ ***Theorem 7: WEAK/ NON-ADAPT|IN(PROD)|OUT(MANY)***

- $\mathscr{K}$ any class of bounded-error quantum circuits such that $\mathrm{post}\mathscr{K}$ contains $\mathbf{PP}$.

- Weak efficient classical simulation of $\mathscr{K} \Rightarrow \mathrm{post}\mathscr{K}$ is contained in $\mathrm{post}\mathbf{BPP}$.

- $\mathrm{post}\mathbf{BPP} \subset \mathbf{PP} \Rightarrow \mathbf{PH}$ would collapse to its third level.