Constant-depth circuits for Uniformly Controlled Gates and Boolean functions with application to quantum memory circuits

https://arxiv.org/abs/2308.08539

Jonathan Allcock[1] Jinge Bao[2] João F. Doriguello[2] Alessandro Luongo[2] Miklos Santha[2]

[1] Tencent [2] Centre for Quantum Technologies

TL;DR

- We show *constant-depth* circuits for
  - UCG: $|x\rangle|\psi\rangle \mapsto |x\rangle U_x|\psi\rangle$
  - Boolean functions: $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$
  - QRAM and QRAG gates
- Bonus: *formal definition of quantum computer with access to quantum memory*

TL;DR

- We show *constant-depth* circuits for
  - UCG: $|x\rangle|\psi\rangle \mapsto |x\rangle U_x|\psi\rangle$
  - Boolean functions: $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$
  - QRAM and QRAG gates
- Bonus: *formal definition of quantum computer with access to quantum memory*

WE CHEAT!

TL;DR

- We show *constant-depth* circuits for
  - UCG: $|x\rangle|\psi\rangle \mapsto |x\rangle U_x|\psi\rangle$
  - Boolean functions: $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$
  - QRAM and QRAG gates
- Bonus: *formal definition of quantum computer with access to quantum memory*

WE CHEAT!

For the complexity theorist: we work in QNC^0_f.

TL;DR

- We show *constant-depth* circuits for
  - UCG: $|x\rangle|\psi\rangle \mapsto |x\rangle U_x|\psi\rangle$
  - Boolean functions: $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$
  - QRAM and QRAG gates
- Bonus: *formal definition of quantum computer with access to quantum memory*

WE CHEAT!

For the complexity theorist: we work in QNC^0_f.

For the experimentalist: we can reduce the depth of many circuits:

$$\log_2(n) \mapsto \log_k(n)$$

Importance of quantum memory

- Data loading in *non-variational* QML algorithms
- Most of HHL-type speedups (with non-sparse matrices)
- State preparation (Grover-Rudolph algorithm)
- Space-time tradeoffs (cryptography)

Importance of quantum memory

- Data loading in *non-variational* QML algorithms
- Most of HHL-type speedups (with non-sparse matrices)
- State preparation (Grover-Rudolph algorithm)
- Space-time tradeoffs (cryptography)

What is a quantum computer with (quantum) access to a memory?

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

- $T \in \mathbb{N}$ is the *depth* of the input to the QPU,
- $|\psi_{\mathtt{I}}\rangle \in \mathtt{I}$,

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

- $T \in \mathbb{N}$ is the *depth* of the input to the QPU,
- $|\psi_{\mathtt{I}}\rangle \in \mathtt{I}$,
- $C_t \in \mathcal{I}(\mathcal{G})$

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

- $T \in \mathbb{N}$ is the *depth* of the input to the QPU,
- $|\psi_{\mathtt{I}}\rangle \in \mathtt{I}$,
- $C_t \in \mathcal{I}(\mathcal{G})$
    - $|\psi_0\rangle := |\psi_{\mathtt{I}}\rangle|0\rangle_{\mathtt{W}} \mapsto |\psi_1\rangle = C_1|\psi_0\rangle \in \mathtt{I} \otimes \mathtt{W}$.

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

- $T \in \mathbb{N}$ is the *depth* of the input to the QPU,
- $|\psi_{\mathtt{I}}\rangle \in \mathtt{I}$,
- $C_t \in \mathcal{I}(\mathcal{G})$
  - $|\psi_0\rangle := |\psi_{\mathtt{I}}\rangle|0\rangle_{\mathtt{W}} \mapsto |\psi_1\rangle = C_1|\psi_0\rangle \in \mathtt{I} \otimes \mathtt{W}$.
  - $\mathcal{I}(\mathcal{G}) =$ all unitaries on $\mathtt{I} \otimes \mathtt{W}$ from $\mathcal{G}$

What is a quantum computer?

Definition: A QPU of size m is defined as a tuple $(\mathtt{I}, \mathtt{W}, \mathcal{G})$ consisting of

- An *input register* $\mathtt{I}$;
- A *workspace* $\mathtt{W}$;
- A constant-size universal gate set $\mathcal{G} \subset \mathcal{U}(\mathbb{C}^{4\times 4})$

An input to the QPU is a tuple $(T, |\psi_{\mathtt{I}}\rangle, C_1, \ldots, C_T)$ where:

- $T \in \mathbb{N}$ is the *depth* of the input to the QPU,
- $|\psi_{\mathtt{I}}\rangle \in \mathtt{I}$,
- $C_t \in \mathcal{I}(\mathcal{G})$
  - $|\psi_0\rangle := |\psi_{\mathtt{I}}\rangle|0\rangle_{\mathtt{W}} \mapsto |\psi_1\rangle = C_1|\psi_0\rangle \in \mathtt{I} \otimes \mathtt{W}$.
  - $\mathcal{I}(\mathcal{G}) =$ all unitaries on $\mathtt{I} \otimes \mathtt{W}$ from $\mathcal{G}$
- The *size* is the sum of the sizes of the instructions $C_1, \ldots, C_T$.

What is a quantum computer with quantum memory?

Definition: A QPU of size poly $\log(n)$ and a Quantum Memory Device (QMD) of n memory registers, where each register is of ℓ-qubit size are collectively defined by a tuple $(\mathtt{I}, \mathtt{W}, \mathtt{A}, \mathtt{T}, \mathtt{Aux}, \mathtt{M}, \mathcal{G}, \mathsf{R})$ consisting of

- Input register, workspace, and gateset (like before, $\mathtt{I}, \mathtt{W}, \mathcal{G}$)
- *Address register* $\mathtt{A}$ ($\log n$-qubits) shared by QPU and QMD;
- A t*arget register* $\mathtt{T}$ (ℓ-qubits) shared by both QPU and QMD;

What is a quantum computer with quantum memory?

Definition: A QPU of size poly $\log(n)$ and a Quantum Memory Device (QMD) of n memory registers, where each register is of ℓ-qubit size are collectively defined by a tuple $(\mathtt{I}, \mathtt{W}, \mathtt{A}, \mathtt{T}, \mathtt{Aux}, \mathtt{M}, \mathcal{G}, \mathsf{R})$ consisting of

- Input register, workspace, and gateset (like before, $\mathtt{I}, \mathtt{W}, \mathcal{G}$)
- *Address register* $\mathtt{A}$ ($\log n$-qubits) shared by QPU and QMD;
- A t*arget register* $\mathtt{T}$ (ℓ-qubits) shared by both QPU and QMD;
- *Auxiliary register* $\mathtt{Aux}$ ($\mathrm{poly}(n)$-qubit) owned by the QMD;

What is a quantum computer with quantum memory?

Definition: A QPU of size $\mathrm{poly}\log(n)$ and a Quantum Memory Device (QMD) of n memory registers, where each register is of ℓ-qubit size are collectively defined by a tuple $(\mathtt{I}, \mathtt{W}, \mathtt{A}, \mathtt{T}, \mathtt{Aux}, \mathtt{M}, \mathcal{G}, \mathsf{R})$ consisting of

- Input register, workspace, and gateset (like before, $\mathtt{I}, \mathtt{W}, \mathcal{G}$)
- *Address register* $\mathtt{A}$ ($\log n$-qubits) shared by QPU and QMD;
- A t*arget register* $\mathtt{T}$ (ℓ-qubits) shared by both QPU and QMD;
- *Auxiliary register* $\mathtt{Aux}$ ($\mathrm{poly}(n)$-qubit) owned by the QMD;
- Memory $\mathtt{M}$ ($n\ell$-qubit) comprising n registers $\mathtt{M}_0, \ldots, \mathtt{M}_{n-1}$, each containing ℓ qubits, owned solely by the QMD;

What is a quantum computer with quantum memory?

Definition: A QPU of size $\mathrm{poly}\log(n)$ and a Quantum Memory Device (QMD) of n memory registers, where each register is of ℓ-qubit size are collectively defined by a tuple $(\mathtt{I}, \mathtt{W}, \mathtt{A}, \mathtt{T}, \mathtt{Aux}, \mathtt{M}, \mathcal{G}, \mathsf{R})$ consisting of

- Input register, workspace, and gateset (like before, $\mathtt{I}, \mathtt{W}, \mathcal{G}$)
- *Address register* $\mathtt{A}$ ($\log n$-qubits) shared by QPU and QMD;
- A t*arget register* $\mathtt{T}$ (ℓ-qubits) shared by both QPU and QMD;
- *Auxiliary register* $\mathtt{Aux}$ ($\mathrm{poly}(n)$-qubit) owned by the QMD;
- Memory $\mathtt{M}$ ($n\ell$-qubit) comprising n registers $\mathtt{M}_0, \ldots, \mathtt{M}_{n-1}$, each containing ℓ qubits, owned solely by the QMD;
- A function $\mathsf{R} : [n] \to \mathcal{V}$, where $\mathcal{V} \subset \mathcal{U}(\mathbb{C}^{2^{2\ell} \times 2^{2\ell}})$ is a $O(1)$-size subset of 2ℓ-qubit gates.

What is a quantum computer with quantum memory?

The instruction set:

$$\mathcal{I}(\mathcal{G},\mathsf{R}) = \mathcal{I}(\mathcal{G}) \cup \{\mathsf{R}\}.$$

where:

$$|i\rangle_{\mathtt{A}}|b\rangle_{\mathtt{T}}|x_i\rangle_{\mathtt{M}_i}|0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)} \mapsto |i\rangle_{\mathtt{A}}\big(\mathsf{R}(i)|b\rangle_{\mathtt{T}}|x_i\rangle_{\mathtt{M}_i}\big)|0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)},$$

What is a quantum computer with quantum memory?

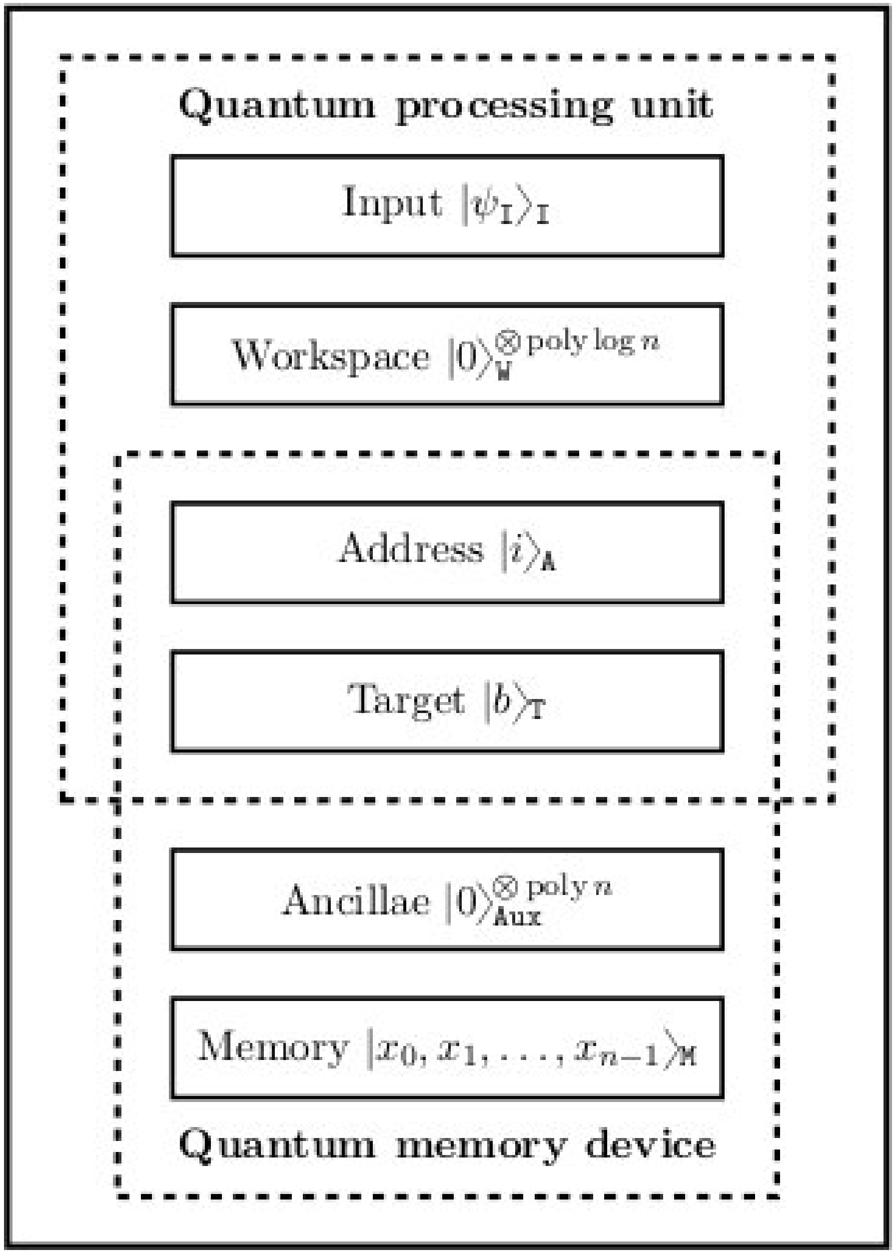
The instruction set:

$$\mathcal{I}(\mathcal{G}, \mathsf{R}) = \mathcal{I}(\mathcal{G}) \cup \{\mathsf{R}\}.$$

where:

$$|i\rangle_{\mathtt{A}} |b\rangle_{\mathtt{T}} |x_i\rangle_{\mathtt{M}_i} |0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)} \mapsto |i\rangle_{\mathtt{A}} \big(\mathsf{R}(i) |b\rangle_{\mathtt{T}} |x_i\rangle_{\mathtt{M}_i}\big) |0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)},$$

In practice?



Quantum processing unit
Input $|\psi_{\mathtt{I}}\rangle_{\mathtt{I}}$
Workspace $|0\rangle_{\mathtt{W}}^{\otimes \operatorname{poly}\log n}$
Address $|i\rangle_{\mathtt{A}}$
Target $|b\rangle_{\mathtt{T}}$
Ancillae $|0\rangle_{\mathtt{Aux}}^{\otimes \operatorname{poly} n}$
Memory $|x_0, x_1, \ldots, x_{n-1}\rangle_{\mathtt{M}}$
Quantum memory device

What we build: Uniformly Controlled Gates

Definition: f-UCG gates:

Consider a function $f : \{0,1\}^n \to \mathcal{U}(\mathbb{C}^{2\times 2})$

$$f\text{-}\mathsf{UCG} = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes f(x) = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes \mathsf{U}_x,$$

What we build: Uniformly Controlled Gates

Definition: f-UCG gates:

Consider a function $f : \{0,1\}^n \to \mathcal{U}(\mathbb{C}^{2\times 2})$

$$f\text{-}\mathsf{UCG} = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes f(x) = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes \mathsf{U}_x,$$

Equivalently, its matrix representation is

$$f\text{-}\mathsf{UCG}^{(n)}_{[n]\to n} = \begin{pmatrix} U_0 & & & \\ & U_1 & & \\ & & \ddots & \\ & & & U_{2^n-1} \end{pmatrix} \in \mathbb{C}^{2^{(n+1)}\times 2^{(n+1)}}.$$

What we build: Uniformly Controlled Gates

Definition: f-UCG gates:

EXAMPLE: HHL

Consider a function $f : \{0,1\}^n \to \mathcal{U}(\mathbb{C}^{2\times 2})$

$$f\text{-}\mathsf{UCG} = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes f(x) = \sum_{x\in\{0,1\}^n} |x\rangle\langle x| \otimes \mathsf{U}_x,$$

Equivalently, its matrix representation is

$$f\text{-}\mathsf{UCG}^{(n)}_{[n]\to n} = \begin{pmatrix} U_0 & & & \\ & U_1 & & \\ & & \ddots & \\ & & & U_{2^n-1} \end{pmatrix} \in \mathbb{C}^{2^{(n+1)}\times 2^{(n+1)}}.$$

What we build: f-FIN gates

Definition: f-FIN gates:

Consider $f : \{0,1\}^{|S|} \mapsto \{0,1\}$

$$f\text{-}\mathsf{FIN}_{S\to i} \left|x_0\right\rangle \left|x_1\right\rangle \ldots \left|x_{m-1}\right\rangle = \left|x_0\right\rangle \ldots \left|x_{i-1}\right\rangle \left|x_i \oplus f(x_S)\right\rangle \left|x_{i+1}\right\rangle \ldots \left|x_{m-1}\right\rangle ,$$

What we build: f-FIN gates

Definition: f-FIN gates:

Consider $f : \{0,1\}^{|S|} \mapsto \{0,1\}$

$$\mathsf{f\text{-}FIN}_{S \to i} \left|x_0\right\rangle \left|x_1\right\rangle \ldots \left|x_{m-1}\right\rangle = \left|x_0\right\rangle \ldots \left|x_{i-1}\right\rangle \left|x_i \oplus f(x_S)\right\rangle \left|x_{i+1}\right\rangle \ldots \left|x_{m-1}\right\rangle ,$$

Equivalently, its matrix representation is

$$f\text{-}\mathsf{FIN}^{(n)}_{[n]\to n} = \begin{pmatrix} X^{f(0)} & & & \\ & {}^{f(1)} & & \\ & & \ddots & \\ & & & X^{f(2^n-1)} \end{pmatrix} \in \mathbb{C}^{2^{(n+1)} \times 2^{(n+1)}}.$$

What we build: f-FIN gates

EXAMPLE:

many things :)

Definition: f-FIN gates:

Consider $f : \{0,1\}^{|S|} \mapsto \{0,1\}$

$$f\text{-FIN}_{S\to i} |x_0\rangle |x_1\rangle \dots |x_{m-1}\rangle = |x_0\rangle \dots |x_{i-1}\rangle |x_i \oplus f(x_S)\rangle |x_{i+1}\rangle \dots |x_{m-1}\rangle ,$$

Equivalently, its matrix representation is

$$f\text{-FIN}^{(n)}_{[n]\to n} = \begin{pmatrix} X^{f(0)} & & & \\ & f(1) & & \\ & & \ddots & \\ & & & X^{f(2^n-1)} \end{pmatrix} \in \mathbb{C}^{2^{(n+1)} \times 2^{(n+1)}}.$$

What we build: QRAM gate

Let $n \in \mathbb{N}$ be a power of 2.

$L = [x_1, \ldots, x_n]^T$ where $x_i \in \{0,1\}^\ell$

Definition: $[\mathsf{QRAM}]$ A *quantum random access memory* of size n is a QMD with $\mathsf{R}(i) = \mathsf{CNOT}_{\mathtt{M}_i \to \mathtt{T}}$,

$$|i\rangle_{\mathtt{A}} |b\rangle_{\mathtt{T}} |x_0, \ldots, x_{n-1}\rangle_{\mathtt{M}} \mapsto |i\rangle_{\mathtt{A}} |b \oplus x_i\rangle_{\mathtt{T}} |x_0, \ldots, x_{n-1}\rangle_{\mathtt{M}}$$

What we build: QRAM gate

Let $n \in \mathbb{N}$ be a power of 2.

$L = [x_1, \ldots, x_n]^T$ where $x_i \in \{0,1\}^\ell$

Definition: [QRAM] A *quantum random access memory* of size n is a QMD with $\mathsf{R}(i) = \mathsf{CNOT}_{\mathtt{M}_i \to \mathtt{T}}$,

$$|i\rangle_{\mathtt{A}}|b\rangle_{\mathtt{T}}|x_0, \ldots, x_{n-1}\rangle_{\mathtt{M}} \mapsto |i\rangle_{\mathtt{A}}|b \oplus x_i\rangle_{\mathtt{T}}|x_0, \ldots, x_{n-1}\rangle_{\mathtt{M}}$$

Memo:

$$|i\rangle_{\mathtt{A}}|b\rangle_{\mathtt{T}}|x_i\rangle_{\mathtt{M}_i}|0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)} \mapsto |i\rangle_{\mathtt{A}}\big(\mathsf{R}(i)|b\rangle_{\mathtt{T}}|x_i\rangle_{\mathtt{M}_i}\big)|0\rangle_{\mathtt{Aux}}^{\otimes \mathrm{poly}(n)},$$

What we build: QRAG gate

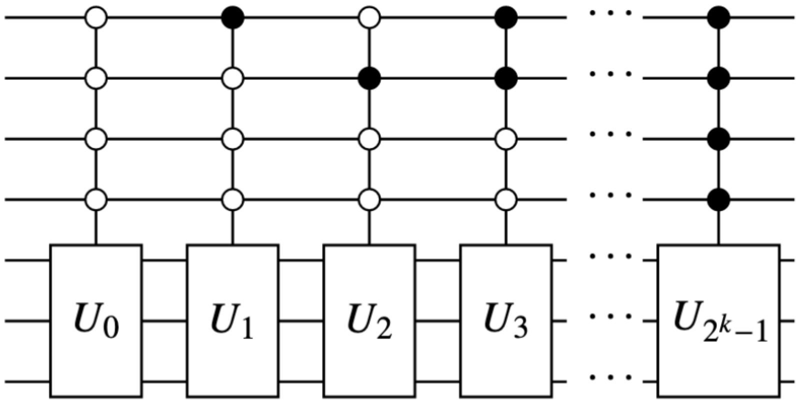
Let $n \in \mathbb{N}$ be a power of 2.

$L = [x_1, \ldots, x_n]^T$ where $x_i \in \{0,1\}^\ell$

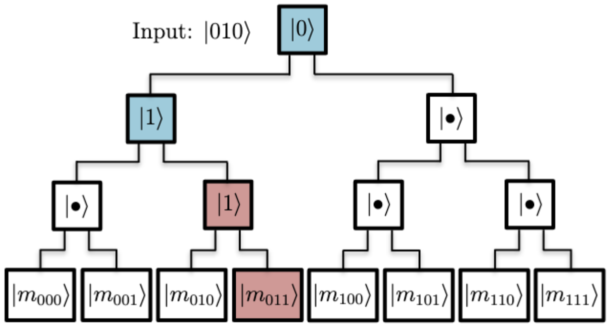
Definition:[QRAG] A *quantum random access gate* of memory size n is a QMD with $\mathsf{R}(i) = \mathsf{SWAP}_{\mathtt{M}_i \leftrightarrow \mathtt{T}}$,

$$|i\rangle_{\mathtt{A}} |b\rangle_{\mathtt{T}} |x_0, \ldots, x_{n-1}\rangle_{\mathtt{M}} \mapsto |i\rangle_{\mathtt{A}} |x_i\rangle_{\mathtt{T}} |x_0, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_{n-1}\rangle_{\mathtt{M}}$$

Some known circuits



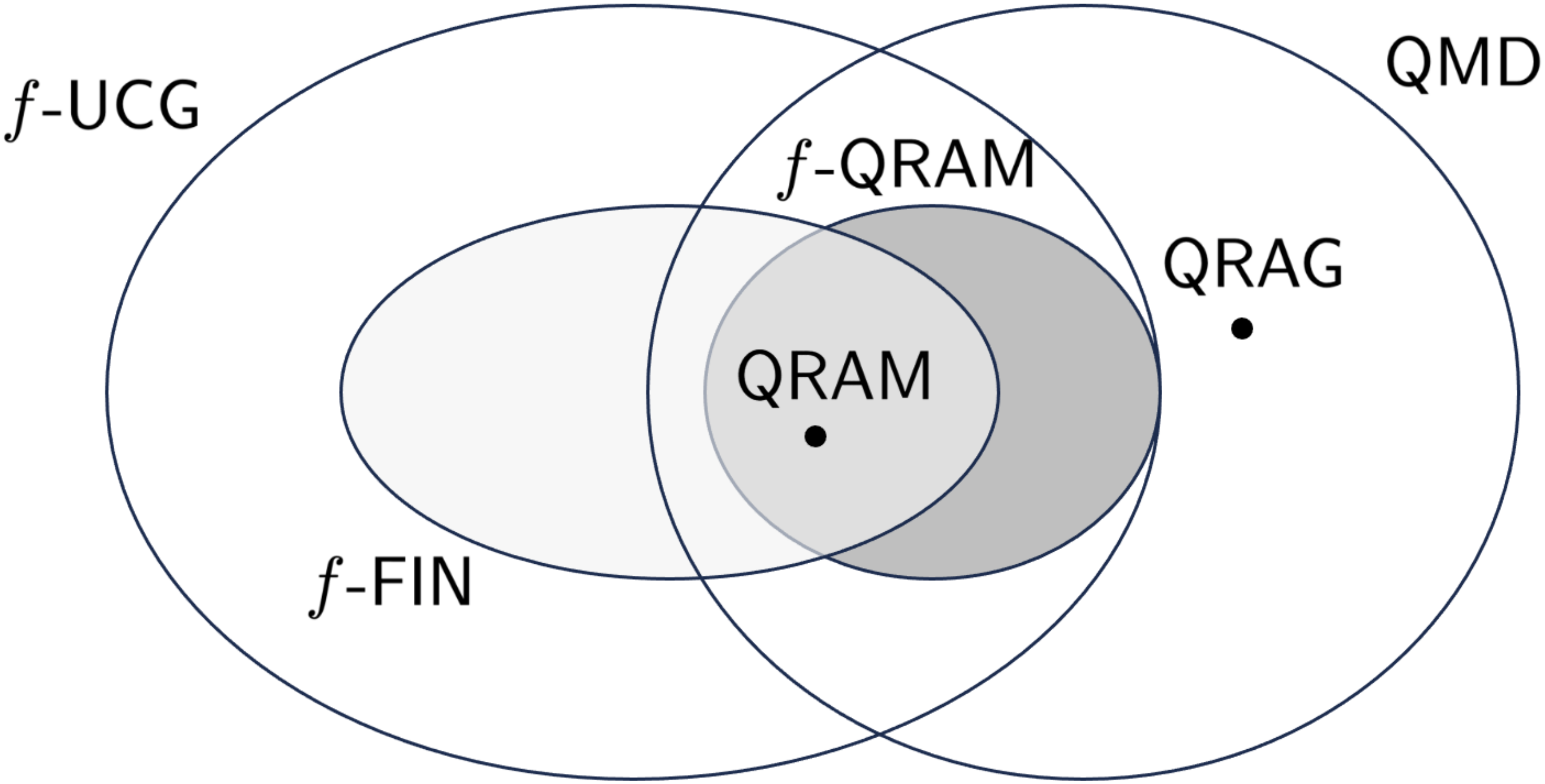
Multiplexer



Bucket-brigade

- Giovannetti, Vittorio, et al. "Architectures for a quantum random access memory." *PRA*
- Liu, Junyu., et al. "Quantum Data Center: Theories and Applications" - https://arxiv.org/pdf/2207.14336.pdf

Images from https://arxiv.org/pdf/2202.11302.pdf and https://arxiv.org/pdf/1502.03450.pdf



f-UCG
QMD
f-QRAM
QRAG
QRAM
f-FIN

Tools and assumptions (1)

Fan-Out gate: It is a sequence of CNOT gates sharing a single control qubit.

$$|b\rangle|x_0,\ldots,x_{n-1}\rangle \mapsto |b\rangle|x_0 \oplus b,\ldots,x_{n-1} \oplus b\rangle.$$

Global-Tunable gate: It is a sequence of CZ gates that can share control or target registers.

Let $\boldsymbol{\Theta} \in [-1,1]^{n\times n}$. The n-arity Global Tunable gate $\mathsf{GT}_{\boldsymbol{\Theta}}^{(n)}$ is the unitary operator

$$\mathsf{GT}_{\boldsymbol{\Theta}}^{(n)} = \prod_{1\leq i<j\leq n} \mathsf{C}_i\mathsf{Z}(\Theta_{ij})_{\to j}.$$

memo: with GT gates we can build Fan-Out gates

Tools and assumptions (2)

Fact: [Z-decomposition of single qubit gates]: for a single-qubit gate U there are angles $\alpha, \beta, \gamma, \delta \in [-1, 1]$ such that

$$\mathsf{U} = e^{i\pi\alpha}\, \mathsf{Z}(\beta)\, \mathsf{H}\, \mathsf{Z}(\gamma)\, \mathsf{H}\, \mathsf{Z}(\delta),$$

Fact: [Equivalence Fan-Out and Parity]: The Fan-Out gate is equivalent to the PARITY up to a Hadamard conjugation.

(i.e. Fan-Out $\Longleftrightarrow$ PARITY)

Tools and assumptions (3)

Theorem: The $\mathsf{AND}^{(n)}$ gate can be implemented in $O(1)$-depth using

- $2n\log n + O(n)$ ancillae and $6n + O(\log n)$ Fan-Out gates with arity at most $2n$.
- $2n + O(\log n)$ ancillae and 4 GT gates with arity at most $n + O(\log n)$.

Claim: A number l of pair-wise commuting Fan-Out gates $\mathsf{FO}^{(n_0)}, \ldots, \mathsf{FO}^{(n_{l-1})}$ can be performed in depth-3 using one GT gate.

T., Yasuhiro, and S. Tani. "Collapse of the hierarchy of constant-depth exact quantum circuits."

B., Sergey, D. Maslov, and Y. Nam. "Constant-Cost Implementations of Clifford Operations and Multiply-Controlled Gates Using Global Interactions."

First construction!

Technique Number 1: convert the input into a *one-hot-encoding*

	Fan-Out Gates		GT Gates	
	#Fan-Out	Ancilla	#GT	Ancilla
f-UCG	$O(n2^n)$	$O(n2^n \log n)$	9	$O(n2^n)$
f-FIN	$O(n2^n)$	$O(n2^n \log n)$	7	$O(n2^n)$
QRAM	$O(n\, log\, n)$	$O(n \log n \log\log n)$	6	$O(n\, log\, n)$

First construction!

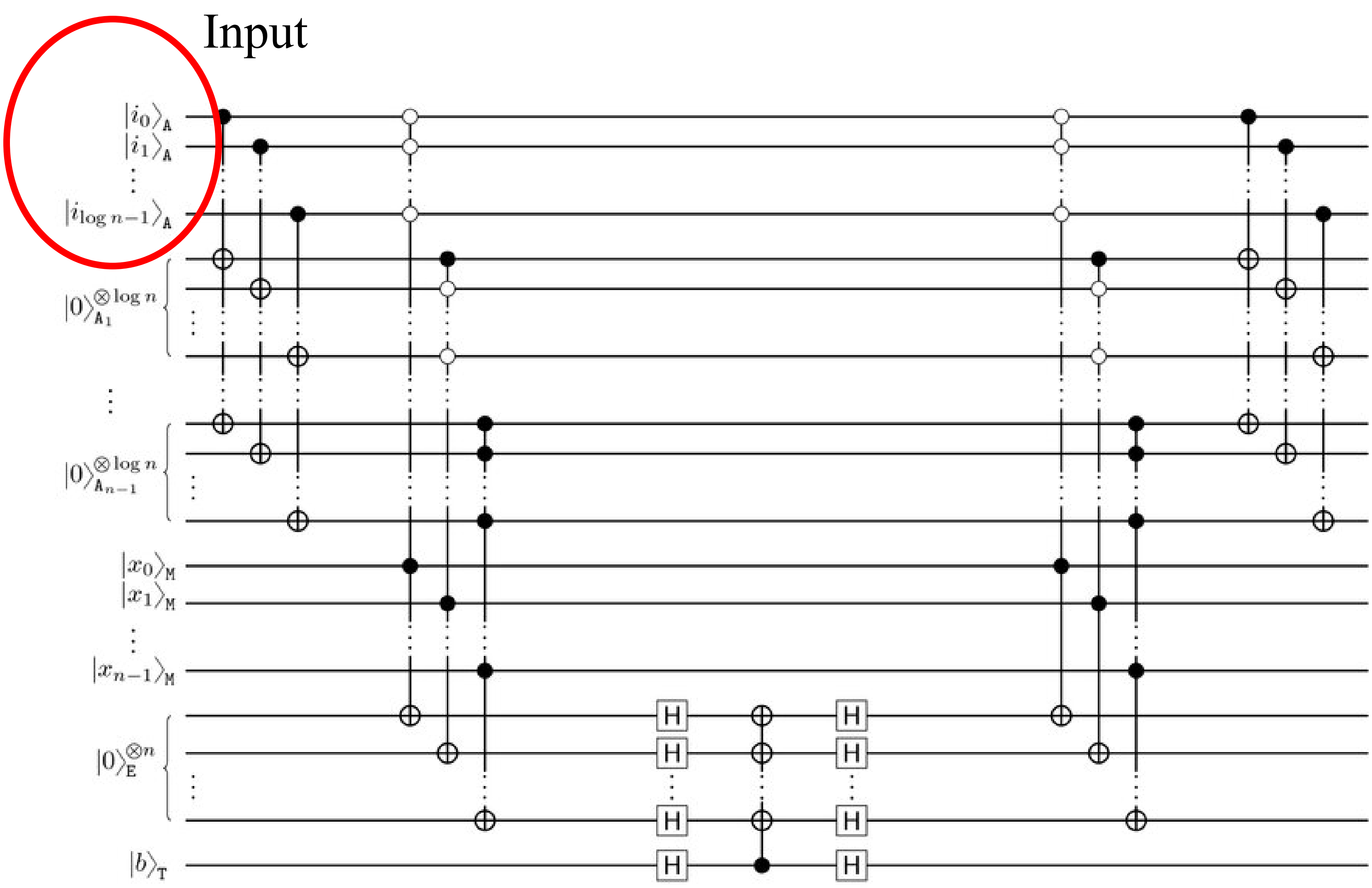
Technique Number 1: convert the input into a *one-hot-encoding*

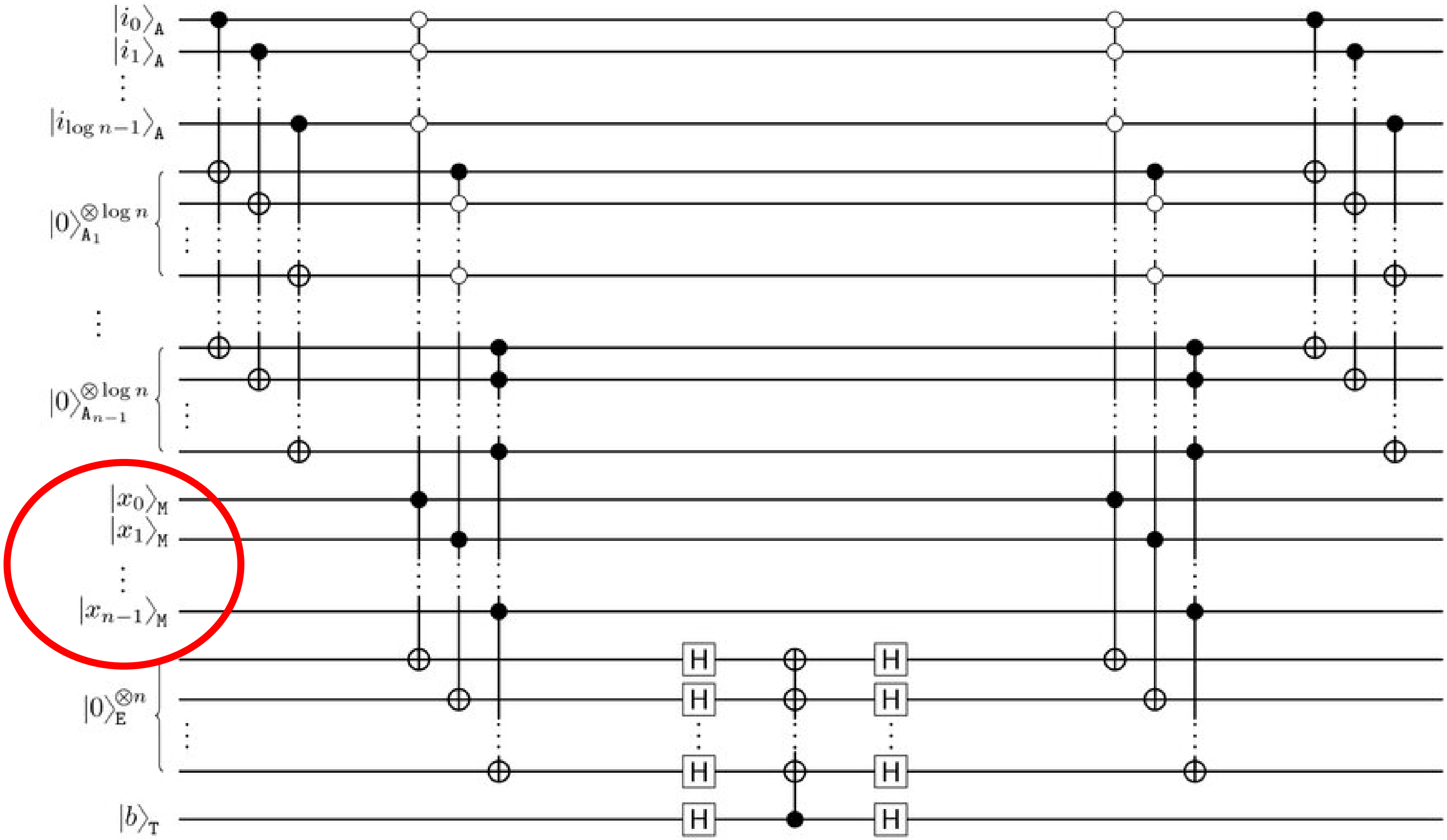
	Fan-Out Gates		GT Gates	
	#Fan-Out	Ancilla	#GT	Ancilla
f-UCG	$O(n2^n)$	$O(n2^n \log n)$	9	$O(n2^n)$
f-FIN	$O(n2^n)$	$O(n2^n \log n)$	7	$O(n2^n)$
QRAM	$O(n \log n)$	$O(n \log n \log \log n)$	6	$O(n \log n)$

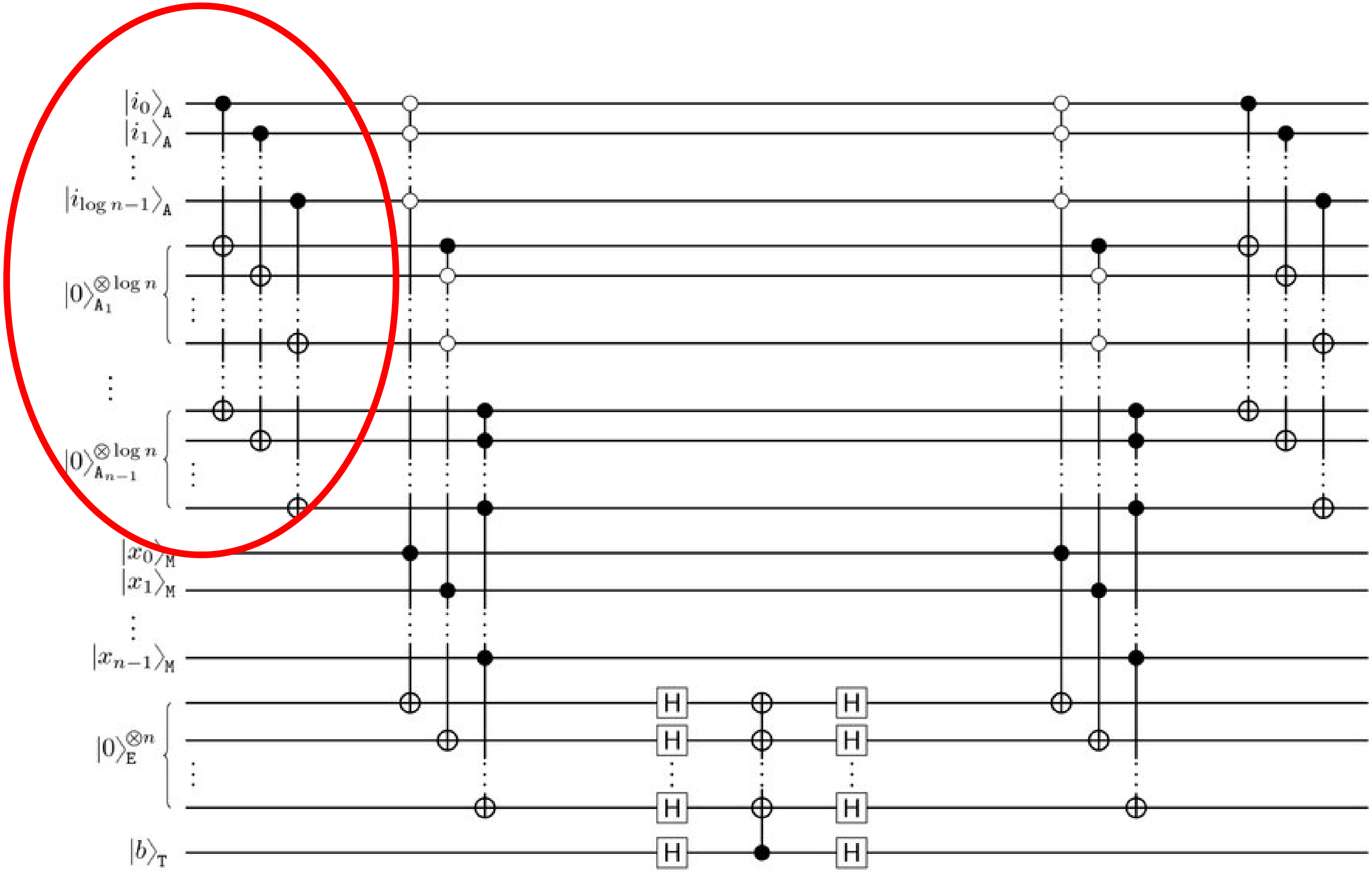
Result	Fan-Out construction		GT construction	
	#Fan-Out	#Ancillae	#GT	#Ancillae
f-UCG $(*)$ [Thm. 26]	$O(n + 2^{t+r}(t+r))$	$O(2^{t+r}(t+r)\log(t+r))$	9	$O(2^{t+r}(t+r))$

Here, f is a (J, r)-junta with $|\overline{J}| = t, t + r \leq n$.

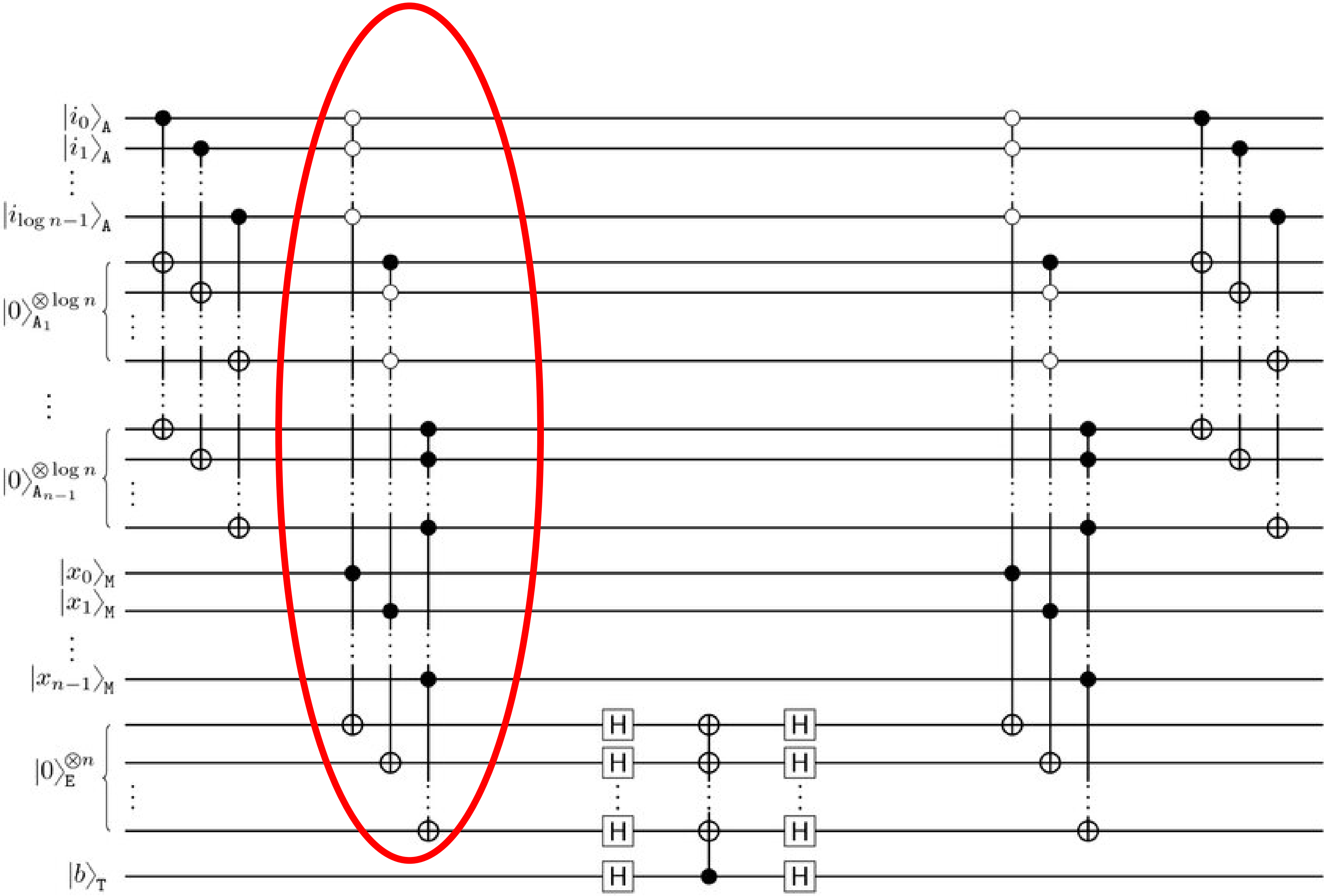
How to build a QRAM in constant depth?



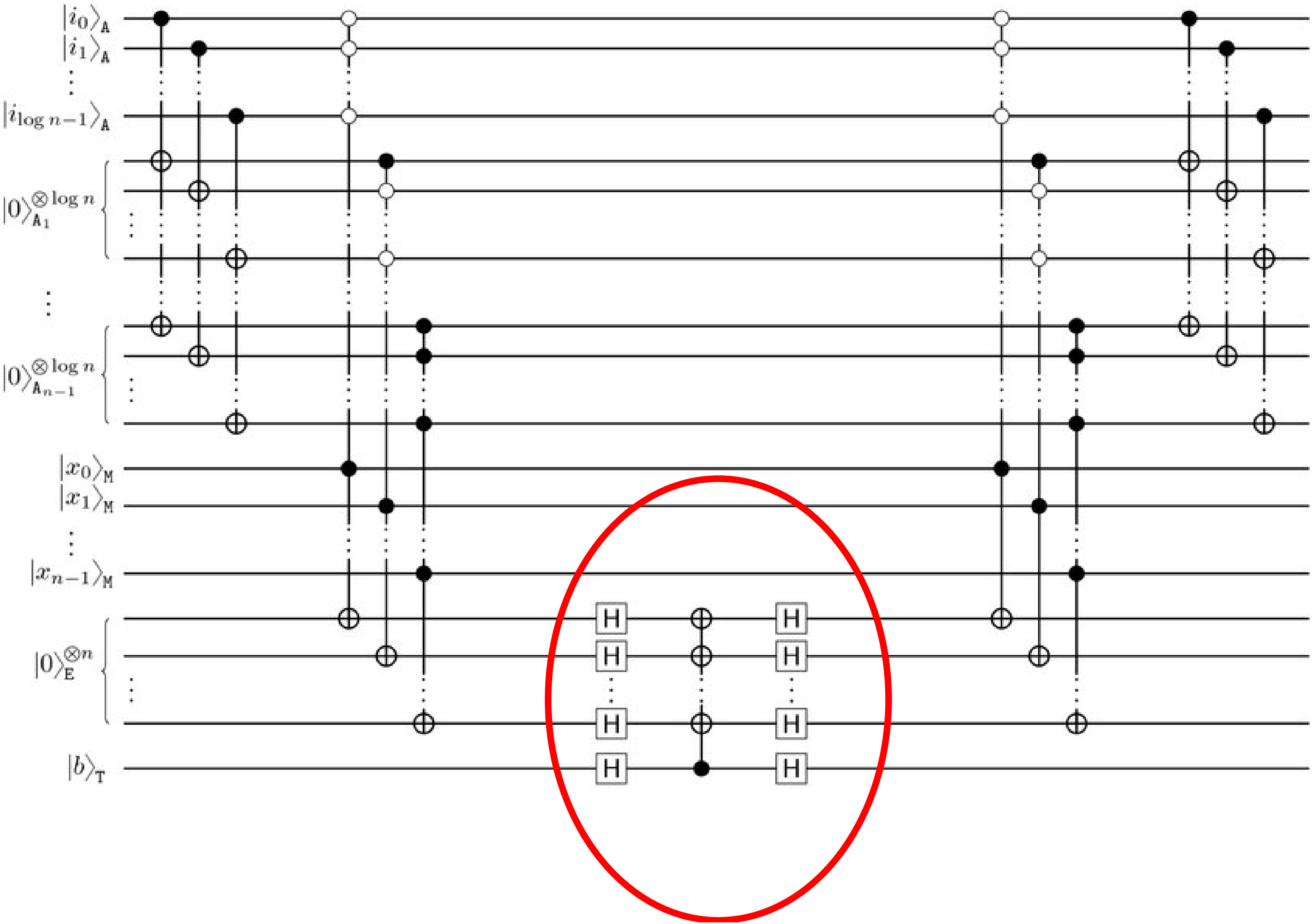

$|i_0\rangle_A$
$|i_1\rangle_A$
$|i_{\log n-1}\rangle_A$
$|0\rangle_{A_1}^{\otimes \log n}$
$|0\rangle_{A_{n-1}}^{\otimes \log n}$
$|x_0\rangle_M$
$|x_1\rangle_M$
$|x_{n-1}\rangle_M$
$|0\rangle_E^{\otimes n}$
$|b\rangle_T$
H



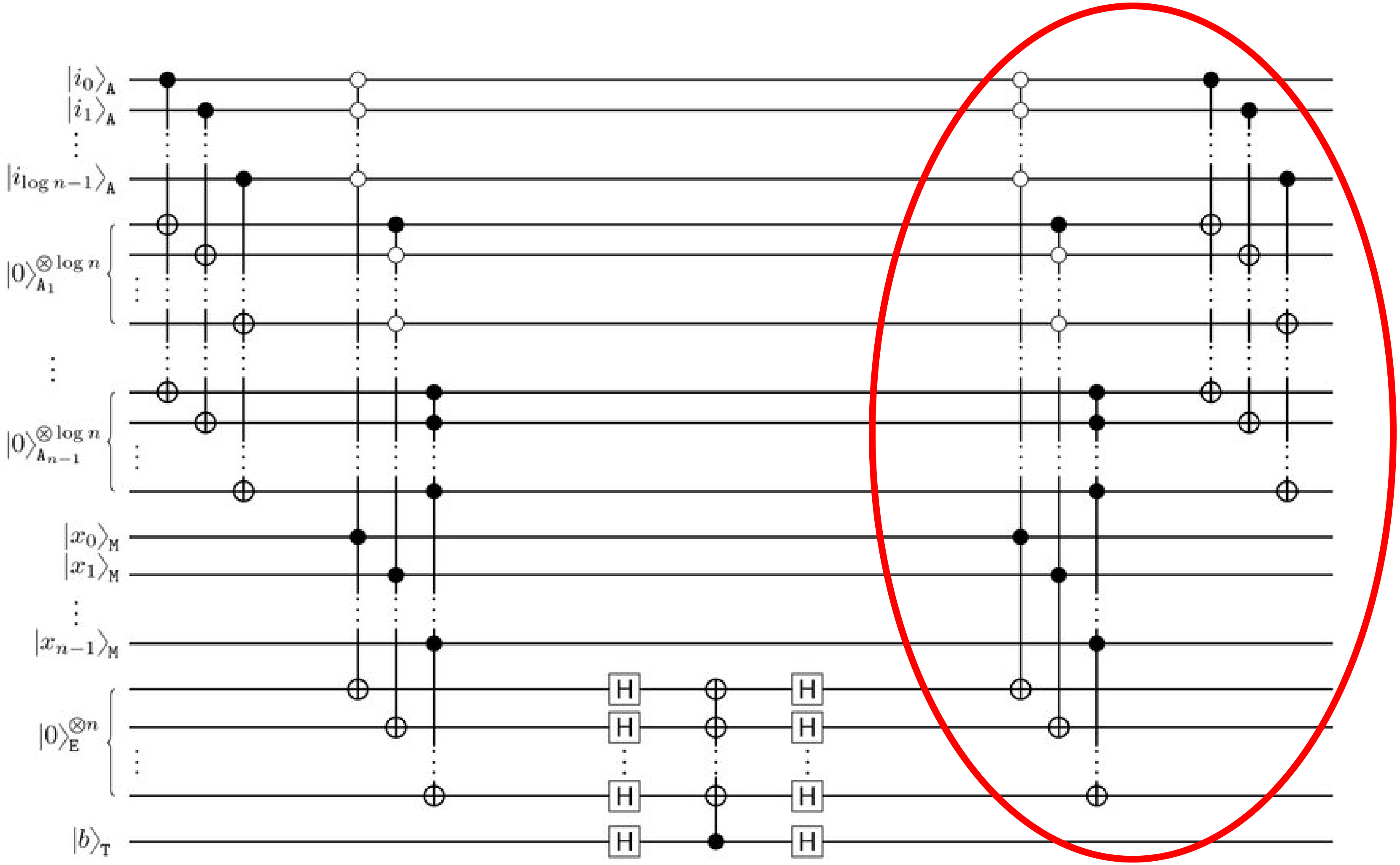
$|i_0\rangle_A$
$|i_1\rangle_A$
$|i_{\log n-1}\rangle_A$
$|0\rangle_{A_1}^{\otimes \log n}$
$|0\rangle_{A_{n-1}}^{\otimes \log n}$
$|x_0\rangle_M$
$|x_1\rangle_M$
$|x_{n-1}\rangle_M$
$|0\rangle_E^{\otimes n}$
$|b\rangle_T$
H



$|i_0\rangle_A$
$|i_1\rangle_A$
$|i_{\log n-1}\rangle_A$
$|0\rangle^{\otimes \log n}_{A_1}$
$|0\rangle^{\otimes \log n}_{A_{n-1}}$
$|x_0\rangle_M$
$|x_1\rangle_M$
$|x_{n-1}\rangle_M$
$|0\rangle^{\otimes n}_E$
$|b\rangle_T$
H

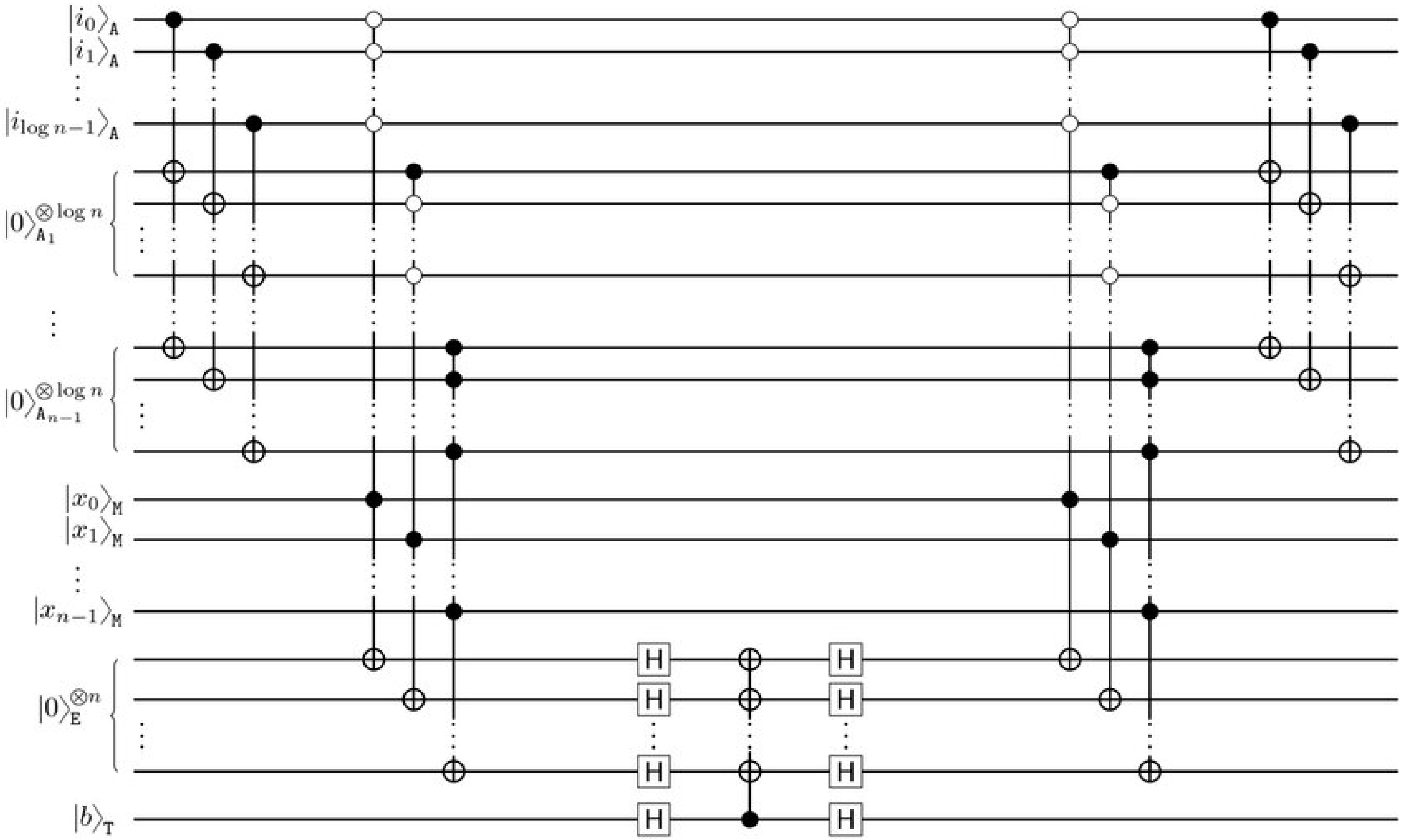


$|i_0\rangle_A$
$|i_1\rangle_A$
$|i_{\log n-1}\rangle_A$
$|0\rangle_{A_1}^{\otimes \log n}$
$|0\rangle_{A_{n-1}}^{\otimes \log n}$
$|x_0\rangle_M$
$|x_1\rangle_M$
$|x_{n-1}\rangle_M$
$|0\rangle_E^{\otimes n}$
$|b\rangle_T$
H



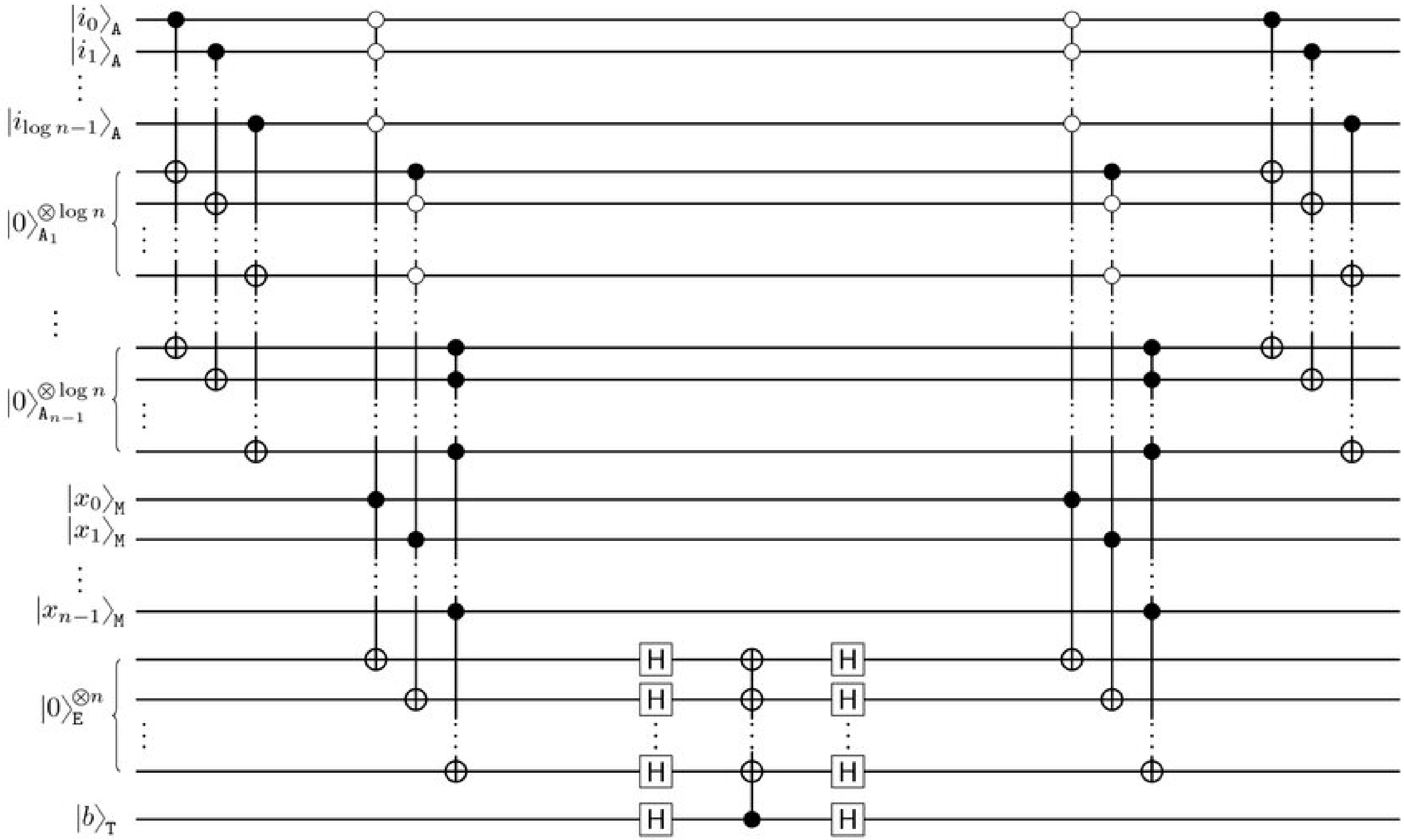
$|i_0\rangle_A$
$|i_1\rangle_A$
$|i_{\log n-1}\rangle_A$
$|0\rangle^{\otimes \log n}_{A_1}$
$|0\rangle^{\otimes \log n}_{A_{n-1}}$
$|x_0\rangle_M$
$|x_1\rangle_M$
$|x_{n-1}\rangle_M$
$|0\rangle^{\otimes n}_E$
$|b\rangle_T$
H

How many GT gates?

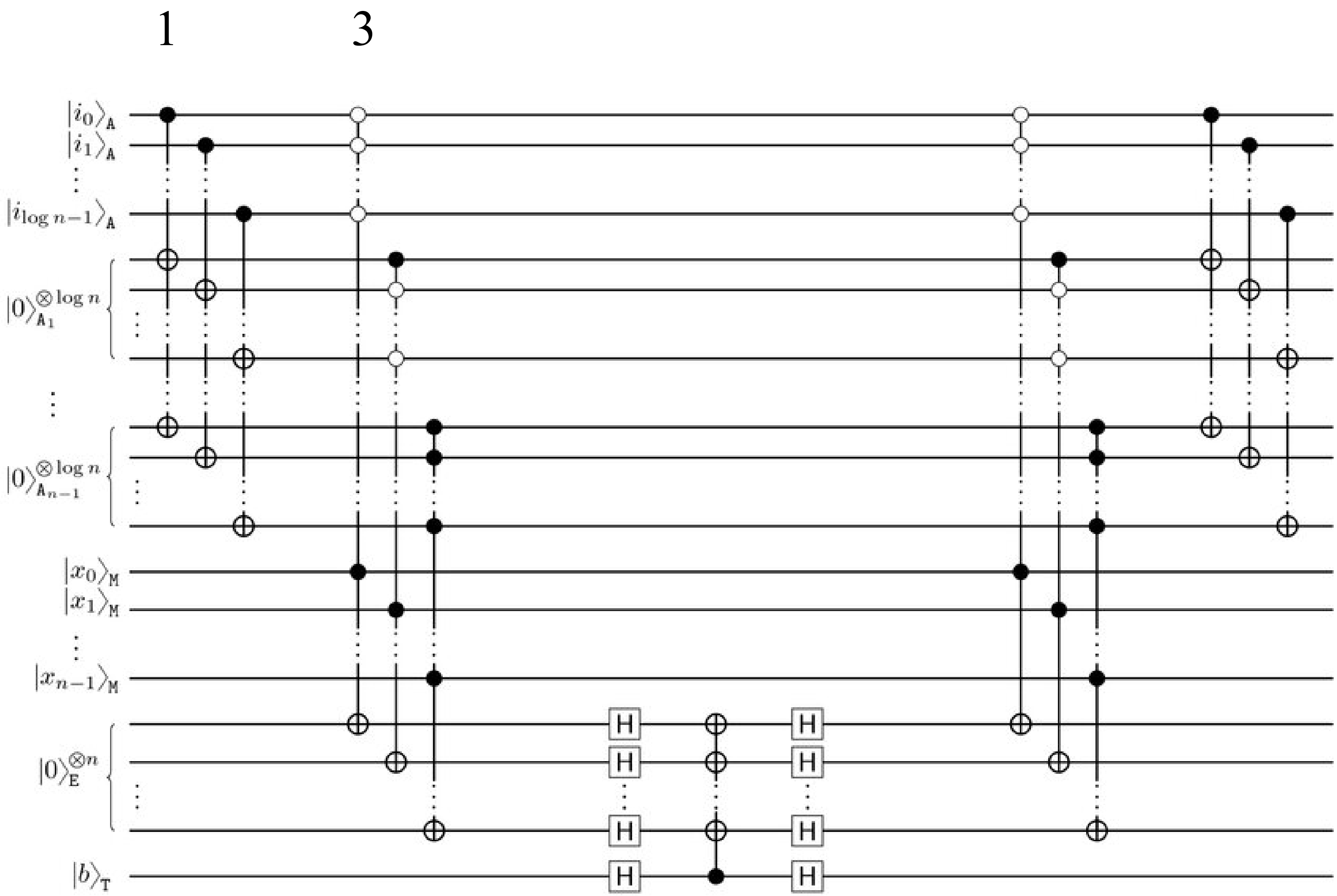


How many GT gates?

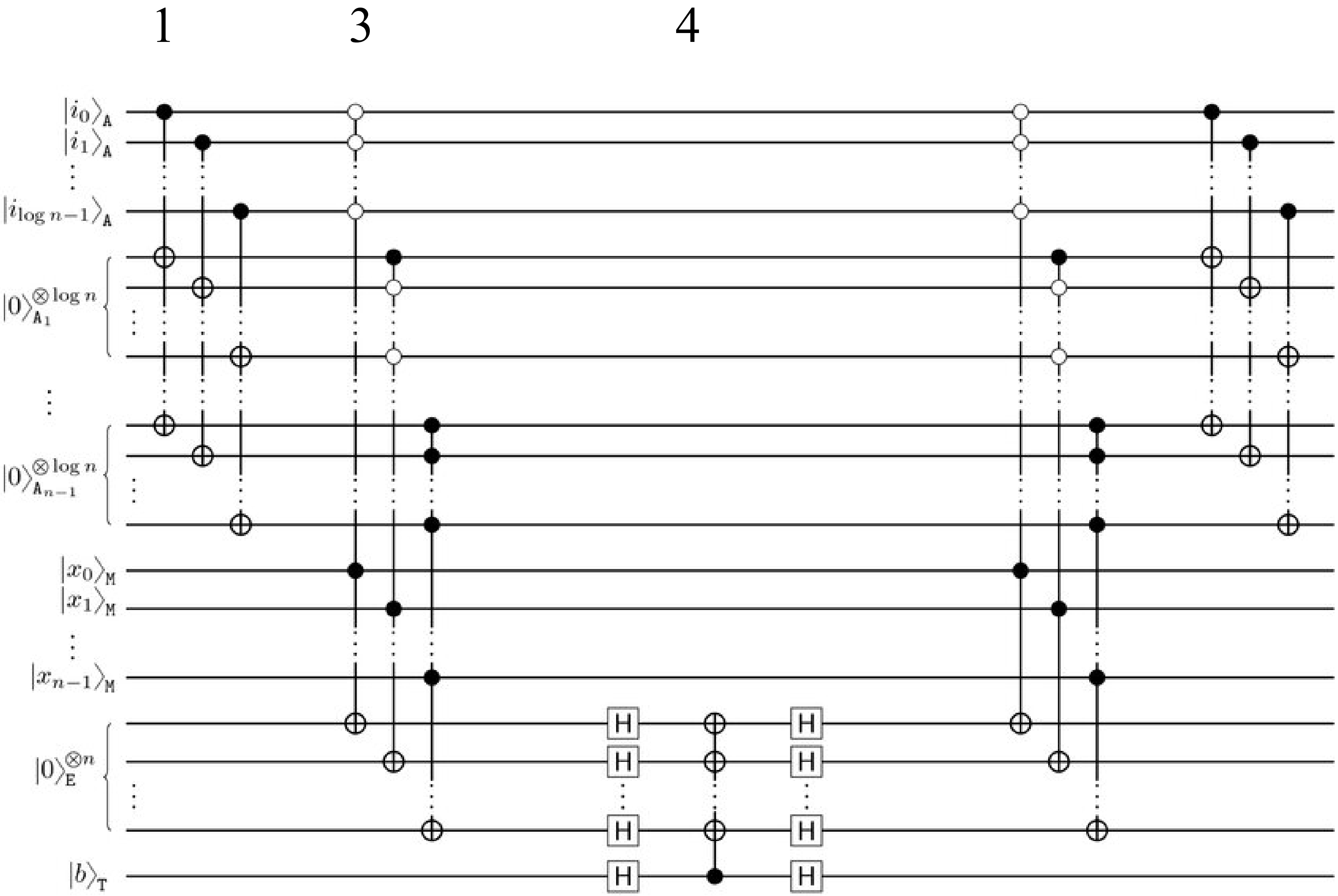
1



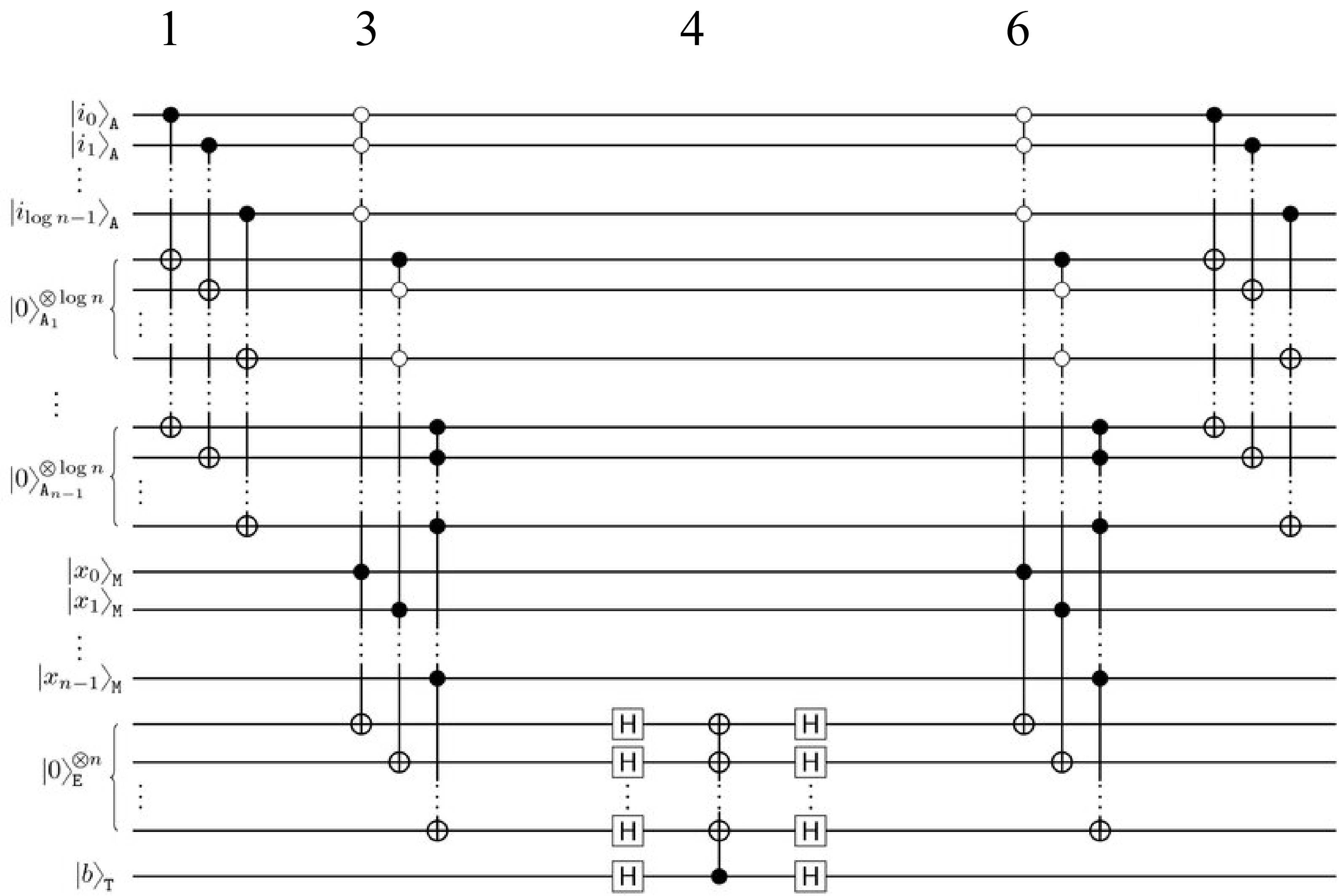
How many GT gates?



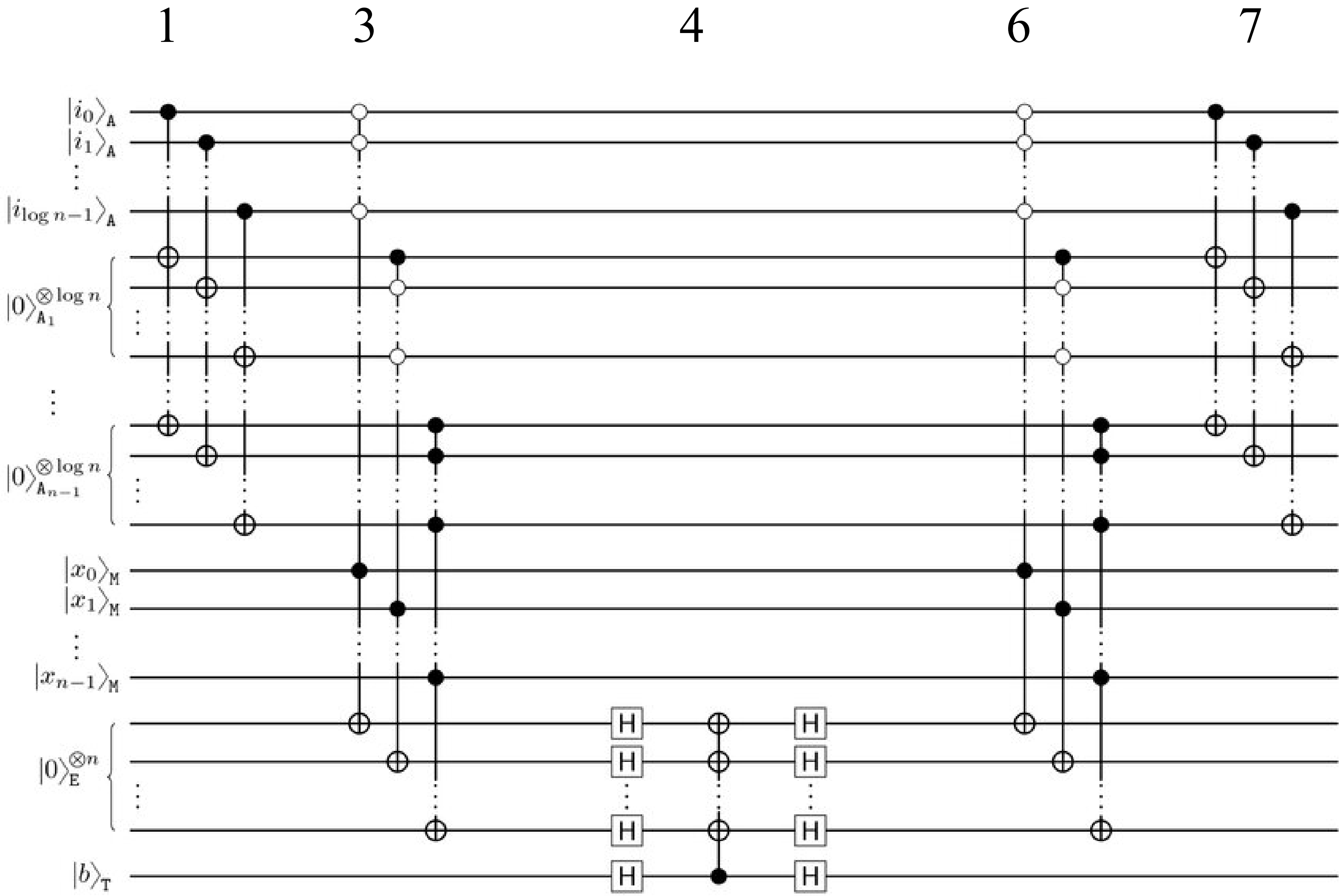
How many GT gates?



How many GT gates?



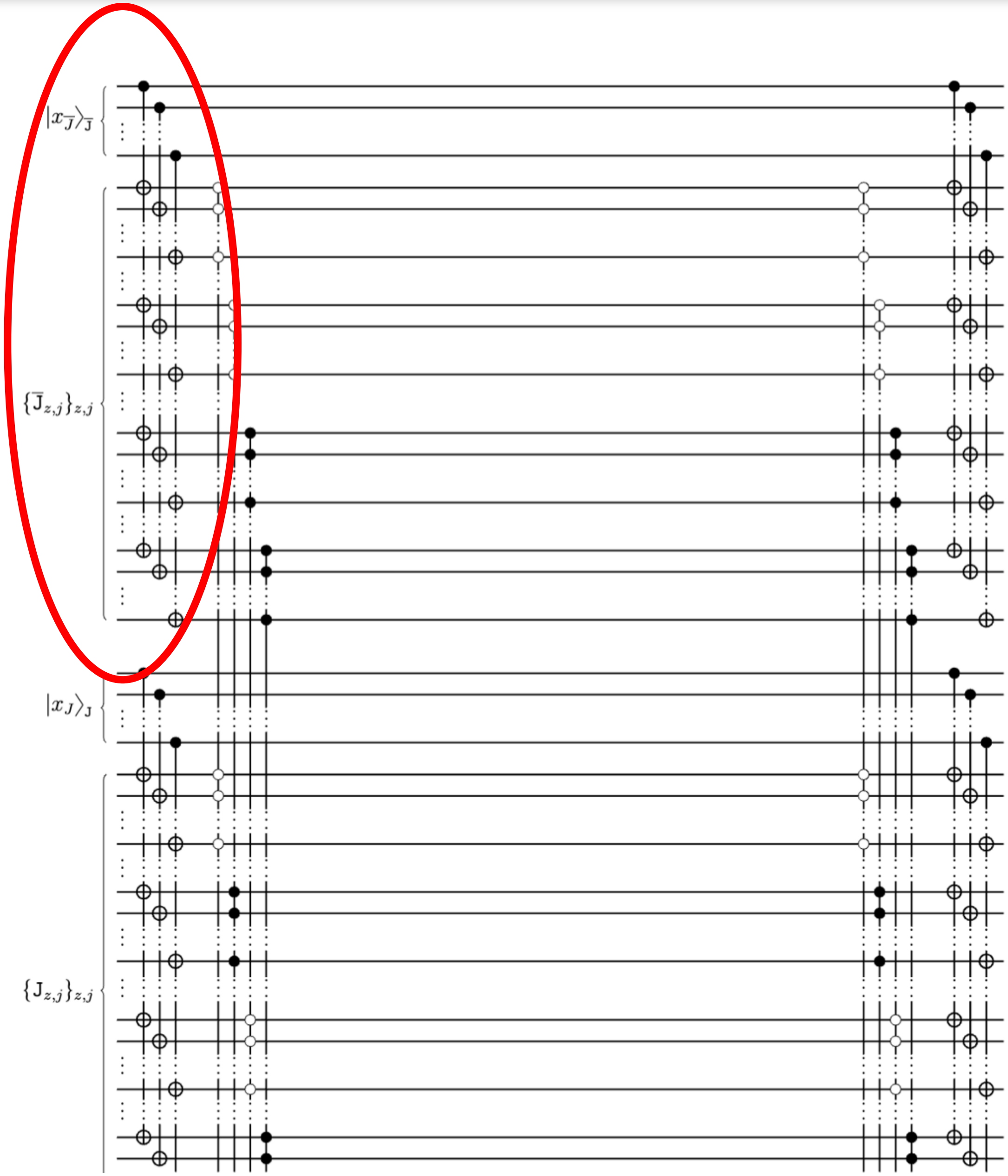
How many GT gates?



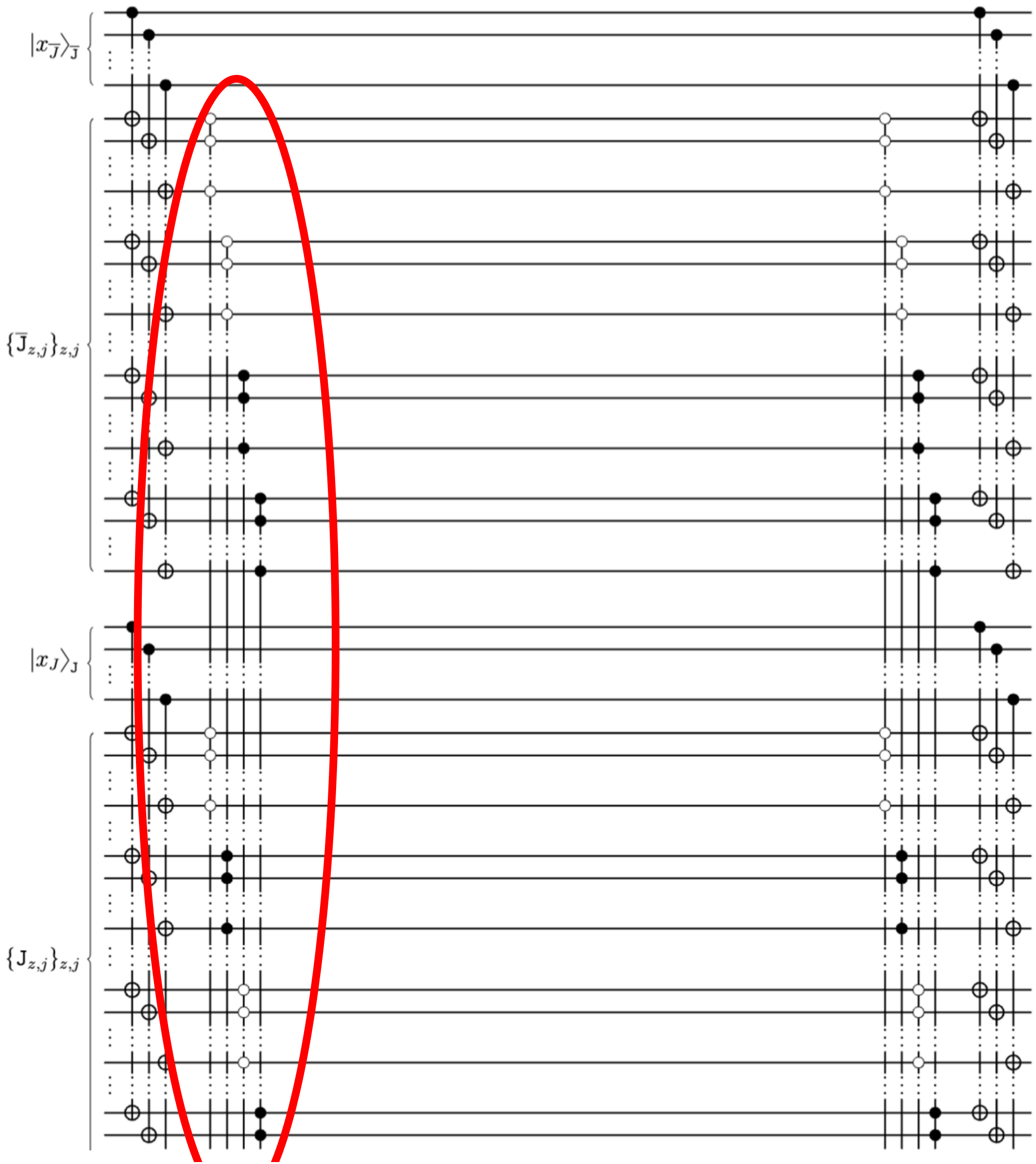
General idea for (J, r)-juntas

- Compute the one-hot encoding of J and $\overline{J}$ separately
- Create copies of the target register
- Apply the Z gates for the Z decomposition of U in parallel
- Undo the copies of the target register
- Undo the computation of the one-hot encoding

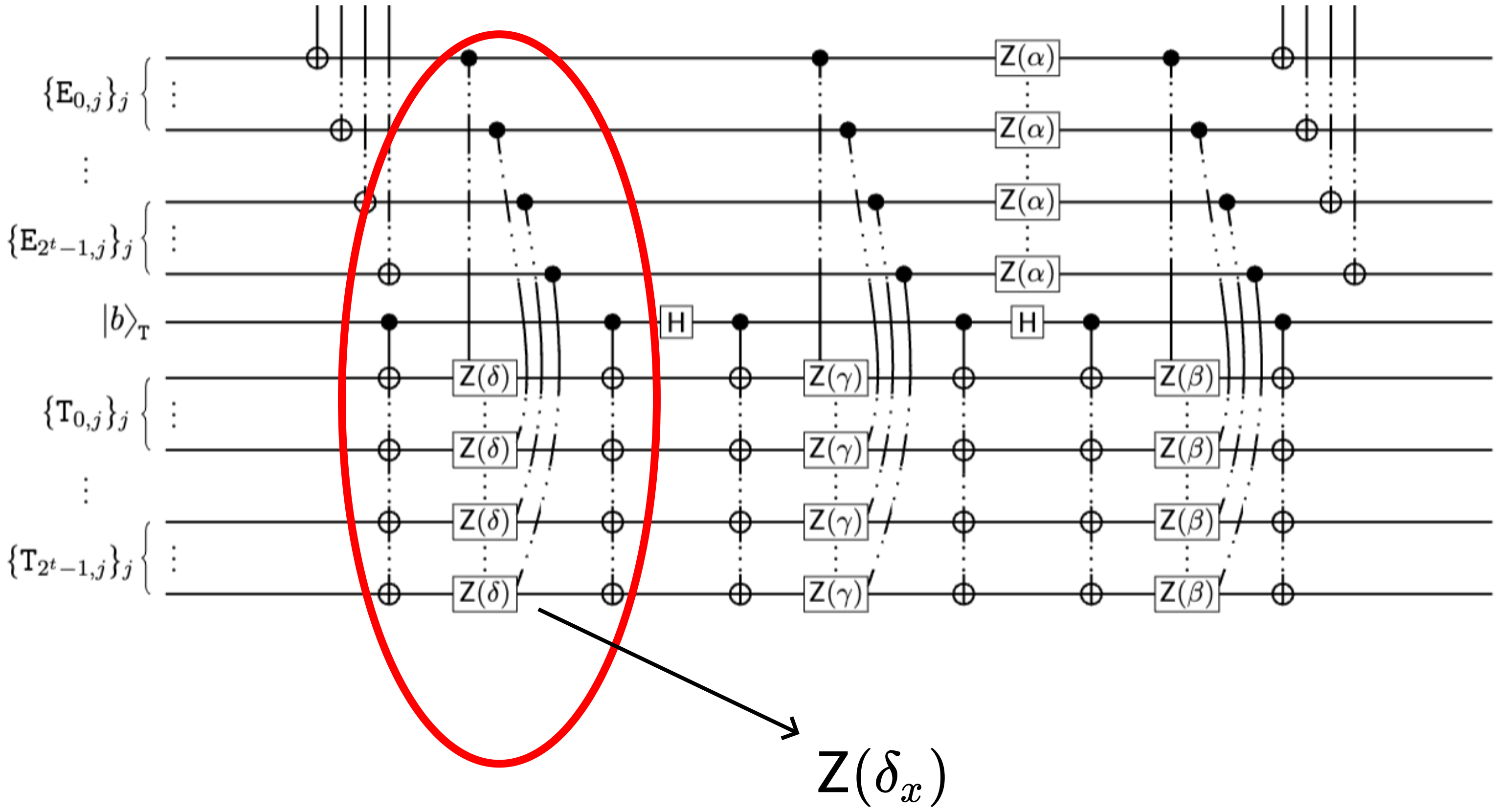
General idea for (J, r)-juntas



General idea for (J, r)-juntas



General idea for (J, r)-juntas



Parallel computation using Fan-Out gates

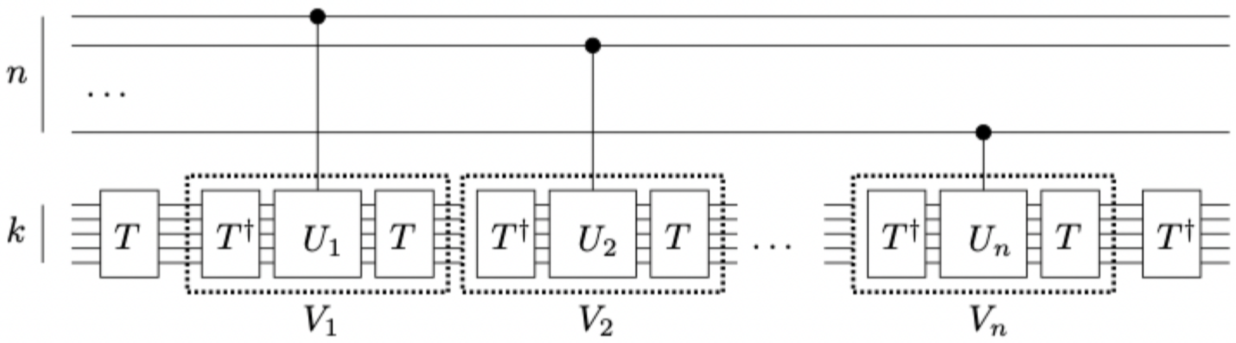


Figure 3: A serial circuit with interpolated basis changes

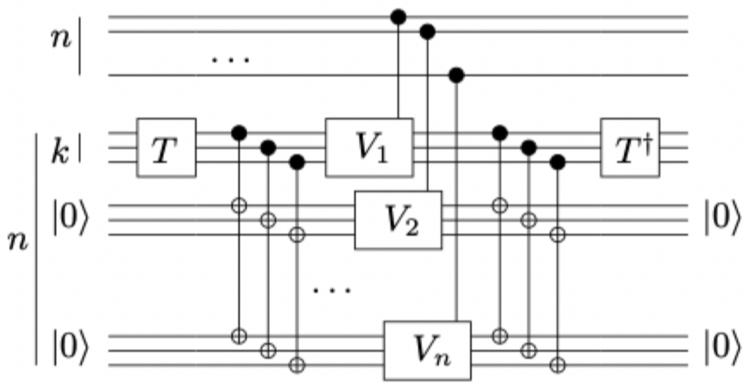


Figure 4: A parallelised circuit performing $U = T^\dagger(\prod_{i=1}^{n} V_i^{x_i})T = \prod_{i=1}^{n} U_i^{x_i}$

Green, F., et al. "Counting, fanout and the complexity of quantum ACC."

Høyer, P., et al. "Quantum fan-out is powerful."

Moore C. et al. "Parallel quantum computation and quantum codes."

Second construction!

Technique Number 2: use good ideas from [1] on the functions obtained by the Z-decomposition of U_x

$$\mathsf{U}_x = e^{i\pi\alpha_x}\,\mathsf{Z}(\beta_x)\,\mathsf{H}\,\mathsf{Z}(\gamma_x)\,\mathsf{H}\,\mathsf{Z}(\delta_x),$$

[1] T., Yasuhiro, and S. Tani. "Collapse of the hierarchy of constant-depth exact quantum circuits"

Second construction!

Technique Number 2: use good ideas from [1] on the functions obtained by the Z-decomposition of U_x

$$\mathsf{U}_x = e^{i\pi\alpha_x}\, \mathsf{Z}(\beta_x)\, \mathsf{H}\, \mathsf{Z}(\gamma_x)\, \mathsf{H}\, \mathsf{Z}(\delta_x),$$

For f-UCG:

[1] T., Yasuhiro, and S. Tani. "Collapse of the hierarchy of constant-depth exact quantum circuits."

Second construction!

Technique Number 2: use good ideas from [1] on the functions obtained by the Z-decomposition of U_x

$$\mathsf{U}_x = e^{i\pi\alpha_x}\, \mathsf{Z}(\beta_x)\, \mathsf{H}\, \mathsf{Z}(\gamma_x)\, \mathsf{H}\, \mathsf{Z}(\delta_x),$$

For f-UCG:

Result	Fan-Out construction		GT construction	
	#Fan-Out	#Ancillae	#GT	#Ancillae
Exact	$\lvert \mathrm{supp}^{>1}(f)\rvert + \lvert \bigcup_{S\in\mathrm{supp}^{>1}(f)} S\rvert$	$\sum_{S\in\mathrm{supp}(f)} \lvert S\rvert$	5	$\lvert \mathrm{supp}^{>1}(f)\rvert$
ϵ-Approximate	$s + \lvert \bigcup_{S\in\mathrm{supp}^{>1}(f)} S\rvert$	$s\deg(f) + \lvert \mathrm{supp}^{=1}(f)\rvert$	5	s
Field $\mathbb{F}_2$	$\sum_{S\in\mathrm{supp}^{>1}_{\{0,1\}}(f)} \lvert S\rvert$	$\sum_{S\in\mathrm{supp}_{\{0,1\}}(f)} \lvert S\rvert \log(1+\lvert S\rvert)$	9	$\sum_{S\in\mathrm{supp}^{>1}_{\{0,1\}}(f)} \lvert S\rvert$

Table 2: Main results for f-UCG for $f : \{0,1\}^n \to \mathcal{U}(\mathbb{C}^{2\times 2})$ Here, $s := (n/\epsilon^2)\sum_{\nu\in\{\alpha,\beta,\gamma,\delta\}} \hat{\|}\nu^{>1}\hat{\|}_1^2$, where $\hat{\|}\nu^{>k}\hat{\|}_1 := \sum_{S\subseteq[n]:|S|>k} |\widehat{\nu}(S)|$, and $\alpha,\beta,\gamma,\delta$ are defined by the Z-decomposition of f; $\mathrm{supp}^{>k}(f) := \{S \subseteq [n] : |S| > k, \widehat{f}(S) \neq 0\}$ is the Fourier support of f with degree greater than k (similarly for $\mathrm{supp}^{=k}(f)$). Big-oh notation is assumed for all entries.

[1] T., Yasuhiro, and S. Tani. "Collapse of the hierarchy of constant-depth exact quantum circuits"

Boolean analysis

First representation is the Fourier expansion (over the reals).

For $g : \{0,1\}^n \mapsto \mathbb{R}$

$$g(x) = \sum_{S \subseteq [n]} \widehat{g}(S) \chi_S(x)$$

- $\widehat{g}(S) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} g(x) \chi_S(x)$, for $S \subseteq [n]$,
- $\chi_S(x) := (-1)^{\sum_{i \in S} x_i}$ is $\mathsf{PARITY}_S(x)$

First representation is the Fourier expansion (over the reals).

For $g : \{0,1\}^n \mapsto \mathbb{R}$

$$g(x) = \sum_{S \subseteq [n]} \widehat{g}(S) \chi_S(x)$$

- $\widehat{g}(S) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} g(x) \chi_S(x)$, for $S \subseteq [n]$,
- $\chi_S(x) := (-1)^{\sum_{i \in S} x_i}$ is $\mathsf{PARITY}_S(x)$
- The Fourier support of g is

$$\mathrm{supp}(g) := \{S \subseteq [n] : \widehat{g}(S) \neq 0\},$$

  while its sparsity is $|\,\mathrm{supp}(f)|$

Boolean analysis

First representation is the Fourier expansion (over the reals).

For $g : \{0,1\}^n \mapsto \mathbb{R}$

$$g(x) = \sum_{S \subseteq [n]} \widehat{g}(S) \chi_S(x)$$

- $\widehat{g}(S) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} g(x) \chi_S(x)$, for $S \subseteq [n]$,
- $\chi_S(x) := (-1)^{\sum_{i \in S} x_i}$ is $\mathsf{PARITY}_S(x)$
- The Fourier support of g is
$$\mathrm{supp}(g) := \{S \subseteq [n] : \widehat{g}(S) \neq 0\},$$
while its sparsity is $|\,\mathrm{supp}(f)|$
- We can build similar definitions for the Z-decomposition of a unitary: $\alpha, \beta, \gamma, \delta : \{0,1\}^n \to [-1,1]$.
  - $\mathrm{supp}(f) := \mathrm{supp}(\alpha) \cup \mathrm{supp}(\beta) \cup \mathrm{supp}(\gamma) \cup \mathrm{supp}(\delta)$
  - $\mathrm{supp}^{>k}(f)$, $\mathrm{supp}^{\leq k}(f)$, $\mathrm{supp}^{=k}(f), \ldots$

More Fourier ideas..

The second representation is based on the existence of a function $p : \{0,1\}^n \to \mathbb{R}$ with a (potentially) sparse Fourier expansion that approximates g: $\epsilon > 0$,

$$\max_{x \in \{0,1\}^n} |p(x) - g(x)| \leq \epsilon$$

.

More Fourier ideas..

The second representation is based on the existence of a function $p : \{0,1\}^n \to \mathbb{R}$ with a (potentially) sparse Fourier expansion that approximates g: $\epsilon > 0$,

$$\max_{x \in \{0,1\}^n} |p(x) - g(x)| \le \epsilon$$

.

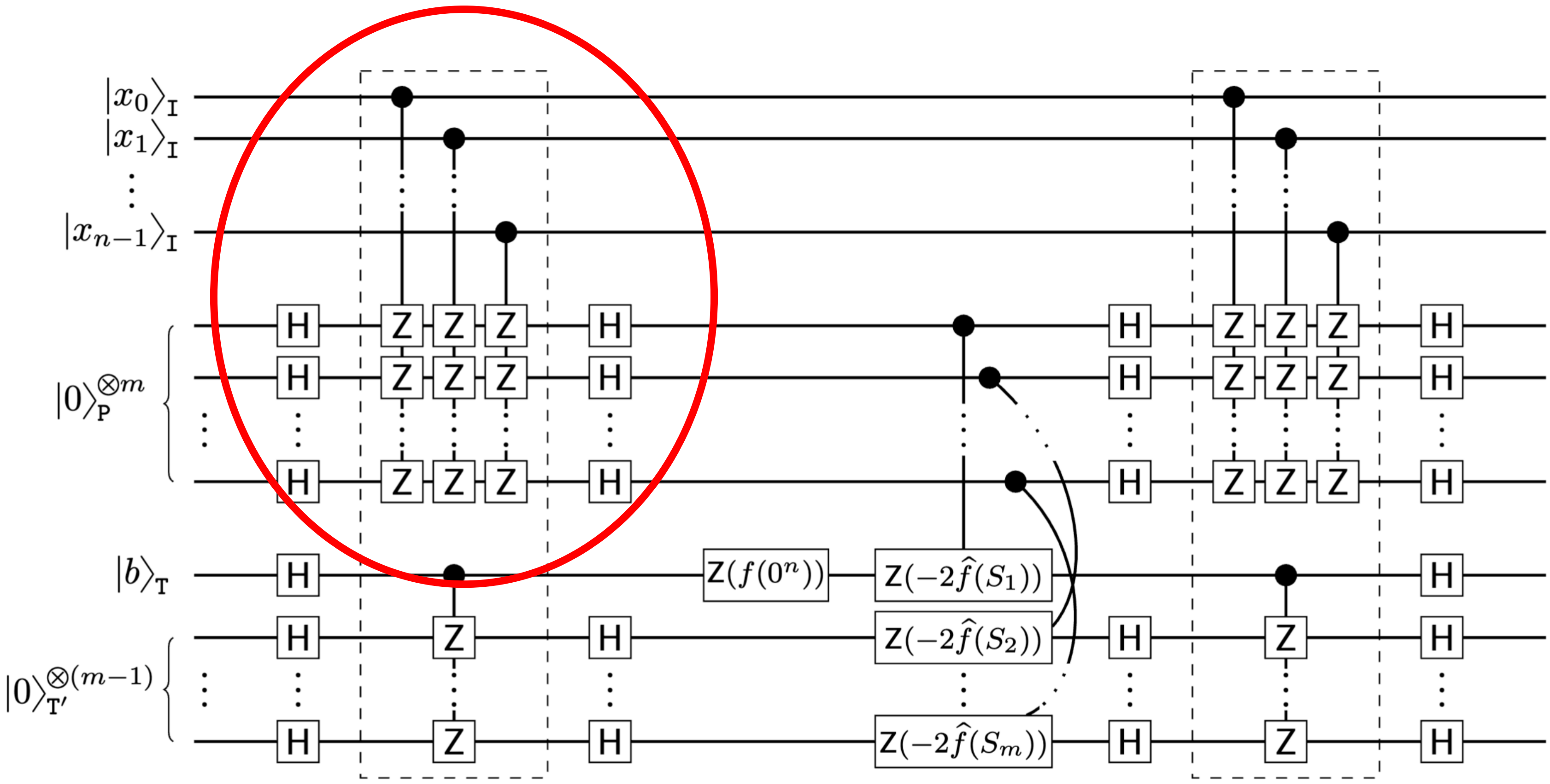
The third representation is using AND functions instead. The (unique) real-polynomial $\{0,1\}$-representation is

$$g(x) = \sum_{S \subseteq [n]} \tilde{g}(S) x^S,$$

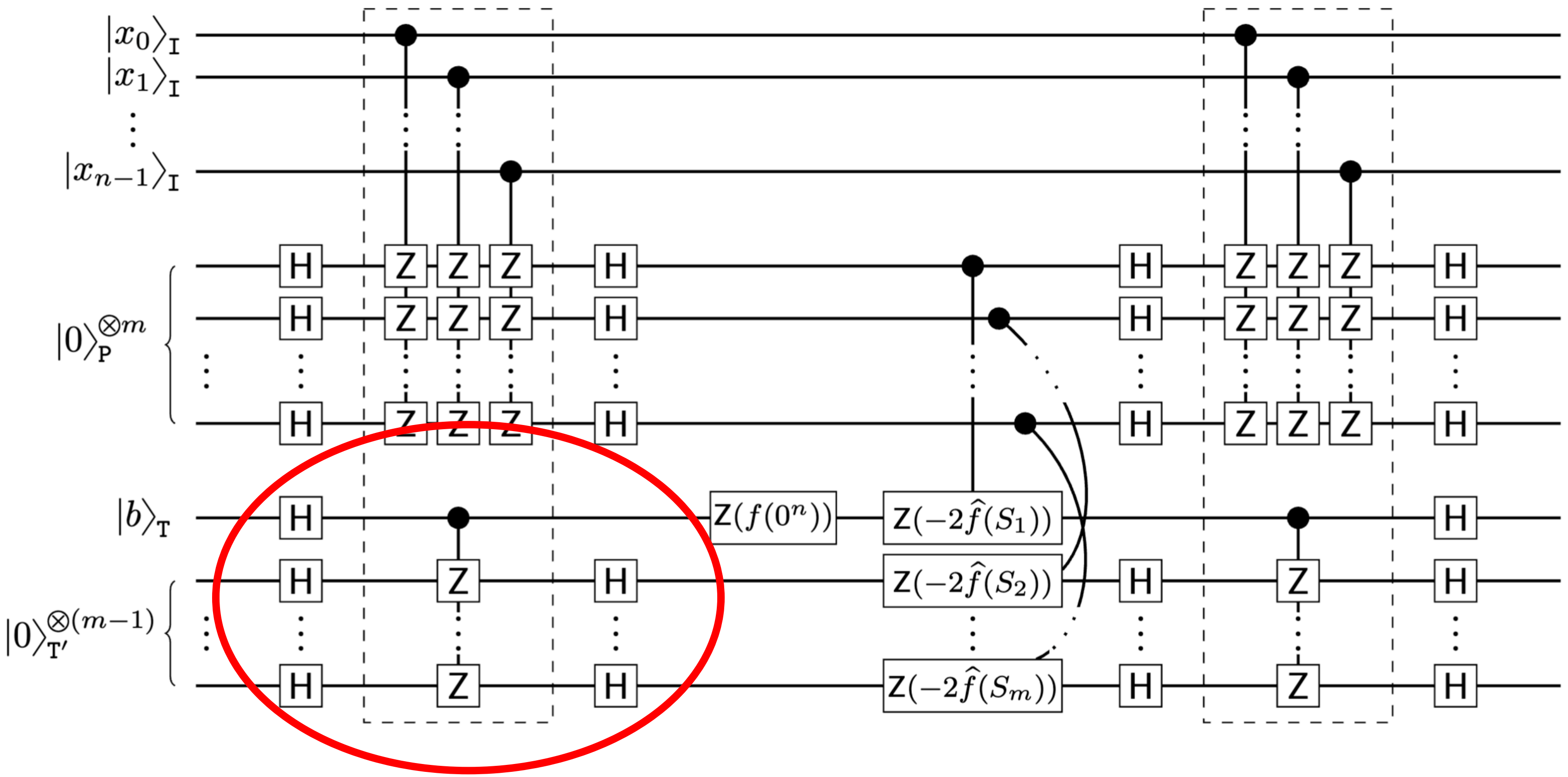
where $x^S := \prod_{i \in S} x_i$ and the coefficients $\tilde{f} : 2^{[n]} \to \mathbb{R}$ are given by $\tilde{f}(S) = \sum_{T \subseteq S} (-1)^{|S|-|T|} f(T)$.

For functions over $\{0,1\}$ we can further change the representation

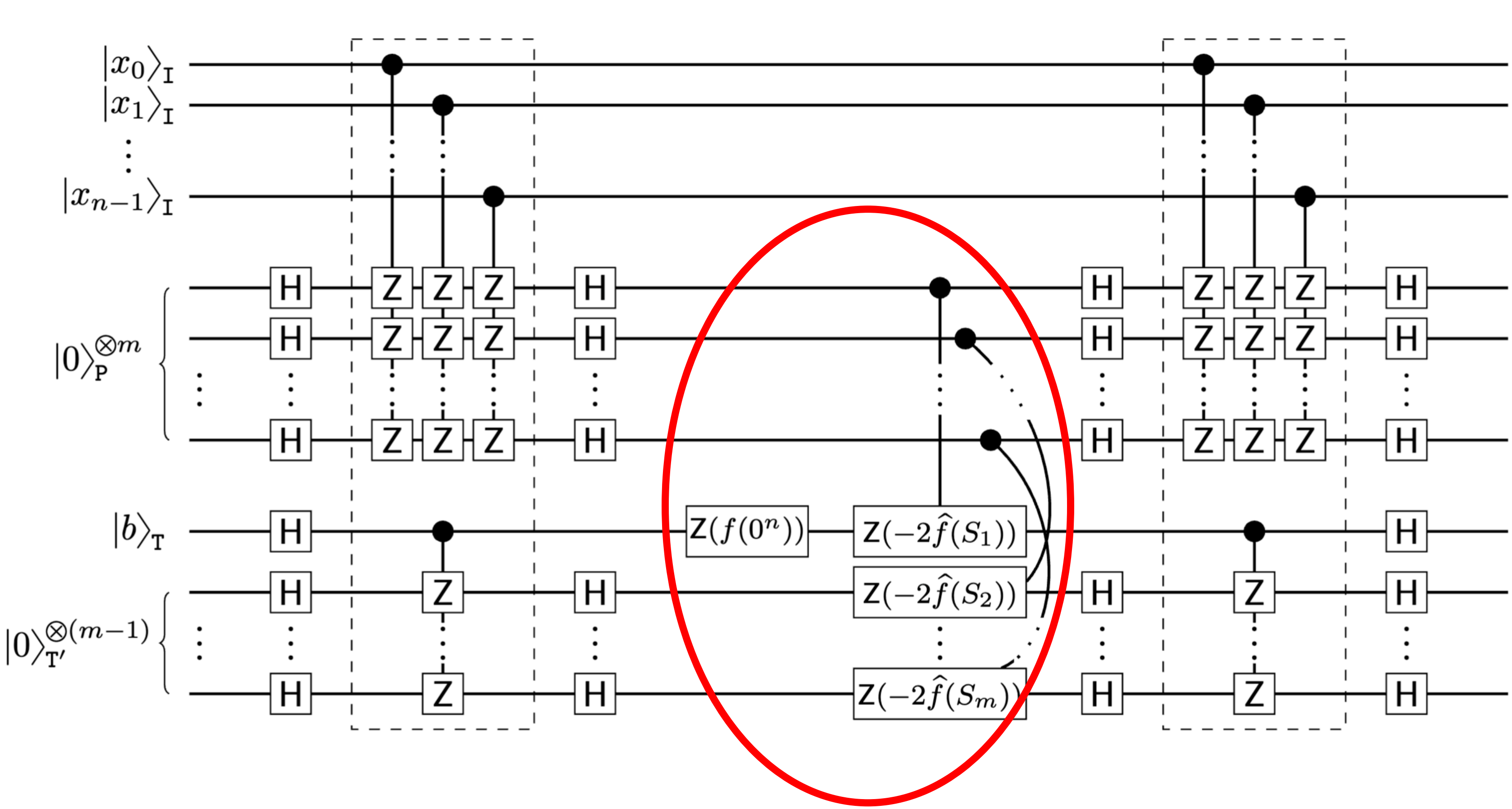
f-FIN gates with Boolean analysis



f-FIN gates with Boolean analysis



f-FIN gates with Boolean analysis



f-UCG: Fourier construction

- **(copy input)** Attach an ancillary register $\bigotimes_{S\in\mathrm{supp}^{>1}(f)} |0\rangle_{\mathtt{R}_S}^{\otimes|S|}$. For each $i\in[n]$ in parallel, copy the qubit $|x_i\rangle_{\mathtt{I}}$ using a Fan-Out gate.
$$|x\rangle_{\mathtt{I}}|b\rangle_{\mathtt{T}} \mapsto |x\rangle_{\mathtt{I}}|b\rangle_{\mathtt{T}} \bigotimes_{S\in\mathrm{supp}^{>1}(f)} |x_S\rangle_{\mathtt{R}_S}.$$
- **(compute parity)** Attach an ancillary register $|0\rangle_{\mathtt{P}}^{\otimes|\mathrm{supp}^{>1}(f)|} = \bigotimes_{S\in\mathrm{supp}^{>1}(f)} |0\rangle_{\mathtt{P}_S}$. For each $S\in\mathrm{supp}^{>1}(f)$ in parallel, apply a $\mathrm{PARITY}^{(|S|)}_{\mathtt{R}_S\to\mathtt{P}_S}$ gate
$$|x\rangle_{\mathtt{I}}|b\rangle_{\mathtt{T}} \bigotimes_{S\in\mathrm{supp}^{>1}(f)} |x_S\rangle_{\mathtt{R}_S} \mapsto |x\rangle_{\mathtt{I}}|b\rangle_{\mathtt{T}} \bigotimes_{S\in\mathrm{supp}^{>1}(f)} |x_S\rangle_{\mathtt{R}_S}\Big|\bigoplus_{i\in S} x_i\Big\rangle_{\mathtt{P}_S}.$$

Fourier construction

- **(copy target)** Copy T for $\mathrm{supp}^{>0}(f)$ times in T′.
- **(apply phase)** For each $S \in \mathrm{supp}^{>0}(\delta)$ in parallel, apply a $\mathsf{Z}(\tilde{\delta}(S))$ gate controlled on register $\mathtt{P}_S$ onto the S-th qubit in register $\mathtt{T}'$
- **(un-copy target)** Undo the copy.
- Observe that the relative phase is summing up, composing δ

$$|x\rangle_{\mathtt{I}}|b\rangle_{\mathtt{T}} \mapsto |x\rangle_{\mathtt{I}}\mathsf{Z}\left(\sum_{S\subseteq[n]}\widehat{\delta}(S)\chi_S(x)\right)|b\rangle^{\otimes m}_{\mathtt{T},\mathtt{T}'} \quad \mapsto |x\rangle_{\mathtt{I}}\mathsf{Z}(\delta(x))|b\rangle_{\mathtt{T}}.$$

f-UCG: Fourier construction

Observe:

$\mathsf{Z}(\delta(x)) = \mathsf{Z}(\sum_{S\in\mathrm{supp}(\delta)} \widehat{\delta}(S)\chi_S(x))!$

Idea: apply $\mathsf{Z}(\delta(x))$ onto a target qubit by simply applying to it a sequence of phases $\mathsf{Z}(\widehat{\delta}(S))$ controlled on $\chi_S(x)$, for $S \in \mathrm{supp}(\delta)$.

f-UCG: Fourier construction

Observe:

$\mathsf{Z}(\delta(x)) = \mathsf{Z}(\sum_{S\in\mathrm{supp}(\delta)} \hat{\delta}(S)\chi_S(x))$!

Idea: apply $\mathsf{Z}(\delta(x))$ onto a target qubit by simply applying to it a sequence of phases $\mathsf{Z}(\hat{\delta}(S))$ controlled on $\chi_S(x)$, for $S \in \mathrm{supp}(\delta)$.

How?

1. First compute $(|0\rangle^{\otimes m} + |1\rangle^{\otimes m})/\sqrt{2}$ from the target qubit using one Fan-Out, where $m := |\,\mathrm{supp}(\delta)|$,
2. Apply the controlled phases $\mathsf{Z}(\hat{\delta}(S))$ onto *different* copies.
   $(|0\rangle^{\otimes m} + (-1)^{\sum_S \hat{\delta}(S)\chi_S(x)}|1\rangle^{\otimes m})/\sqrt{2} = \mathsf{Z}(\delta(x))(|0\rangle^{\otimes m} + |1\rangle^{\otimes m})/\sqrt{2}$
3. Uncompute the copies with another Fan-Out.

QRAM as a function

Let $f : \{0,1\}^n \times \{0,1\}^{\log n} \mapsto \{0,1\}$.

The QRAM function is defined as $f(x,i) = x_i$.

QRAM as a function

Let $f : \{0,1\}^n \times \{0,1\}^{\log n} \mapsto \{0,1\}$.

The QRAM function is defined as $f(x,i) = x_i$.

We can do Fourier analysis on this function!

QRAM as a function

Let $f : \{0,1\}^n \times \{0,1\}^{\log n} \mapsto \{0,1\}$.

The QRAM function is defined as $f(x, i) = x_i$.

We can do Fourier analysis on this function!

Theorem: Let $n \in \mathbb{N}$ be a power of 2. A QRAM of memory size n can be implemented in constant-depth using

- either $\frac{1}{2}n^2 \log n$ ancillae and $2n^2$ Fan-Out gates,
- or $2n^2$ ancillae and 2 GT gates.

Bonus slide: QRAG vs QRAM

- **Theorem:** A query to a QRAM of memory size n can be simulated using 2 queries to a QRAG of memory size n,

Bonus slide: QRAG vs QRAM

- **Theorem:** A query to a QRAM of memory size n can be simulated using 2 queries to a QRAG of memory size n,
- **Theorem:** In our computational model, a query to a QRAG **cannot** be simulated by any number of queries to a QRAM.

Bonus slide: QRAG vs QRAM

- **Theorem:** A query to a QRAM of memory size n can be simulated using 2 queries to a QRAG of memory size n,
- **Theorem:** In our computational model, a query to a QRAG **cannot** be simulated by any number of queries to a QRAM.
- **Theorem:** ... but suppose that single-qubit gates can be freely applied onto the memory register `M` of any QRAM. Then a QRAG of memory size n can be simulated using 3 queries to a QRAM of memory size n and $2(n+1)$ Hadamard gates.

Conclusions

- We show *constant-depth* circuits:
  - UCGs
  - Boolean functions
  - quantum memory gates
- We can improve the depth of many circuits:

$$\log_2(n) \mapsto \log_k(n)$$

- Bonus: *formal definition of quantum computer with access to quantum memory*
- **Future work? Moral of the story?**

Centre for Quantum Technologies

Open Source lecture notes:

QUANTUMALGORITHMS.ORG

PhDs/postdocs:

cs@quantumlah.org