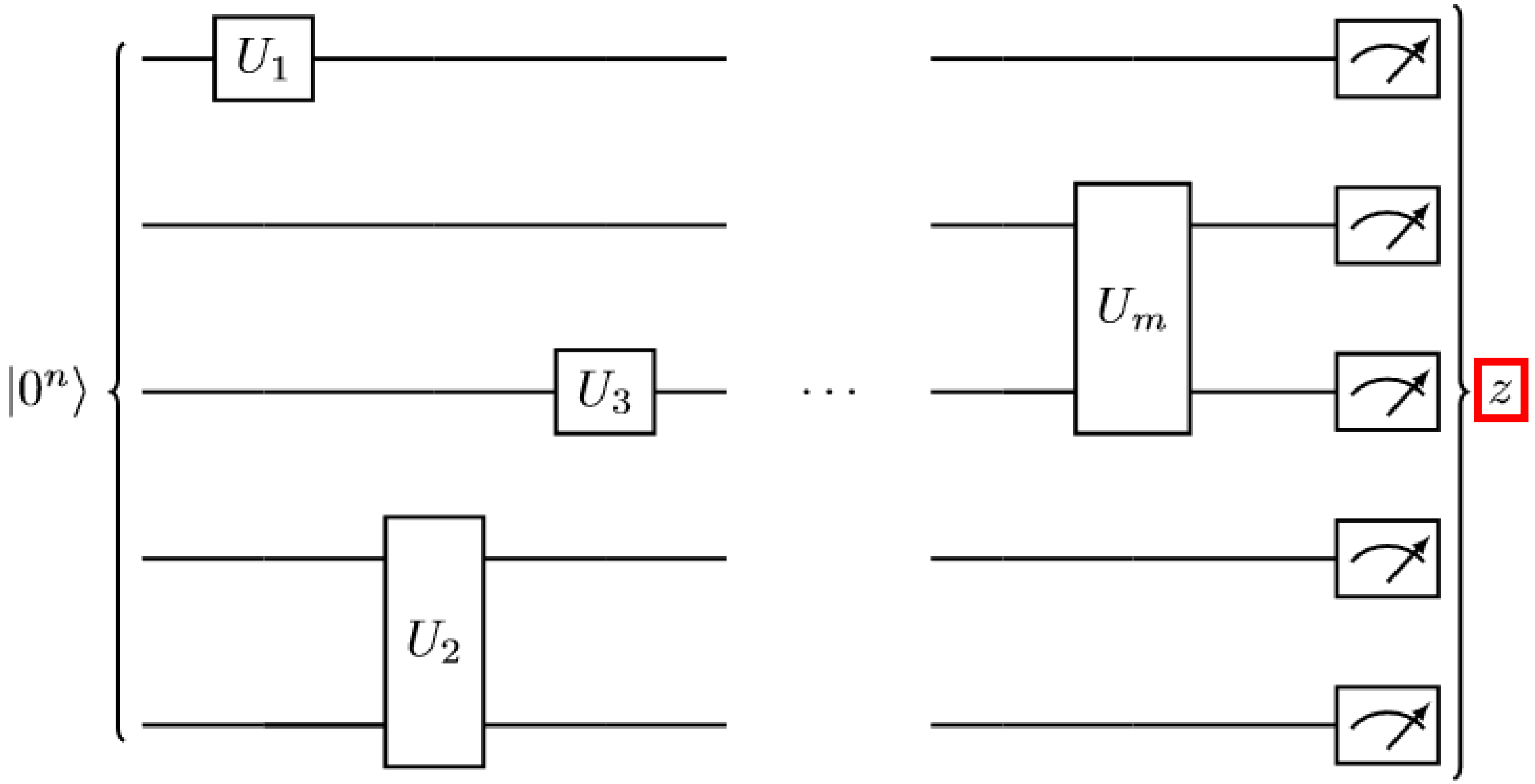
Fast simulation of planar Clifford circuits

David Gosset Daniel Grier **Alex Kerzner** Luke Schaeffer

UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

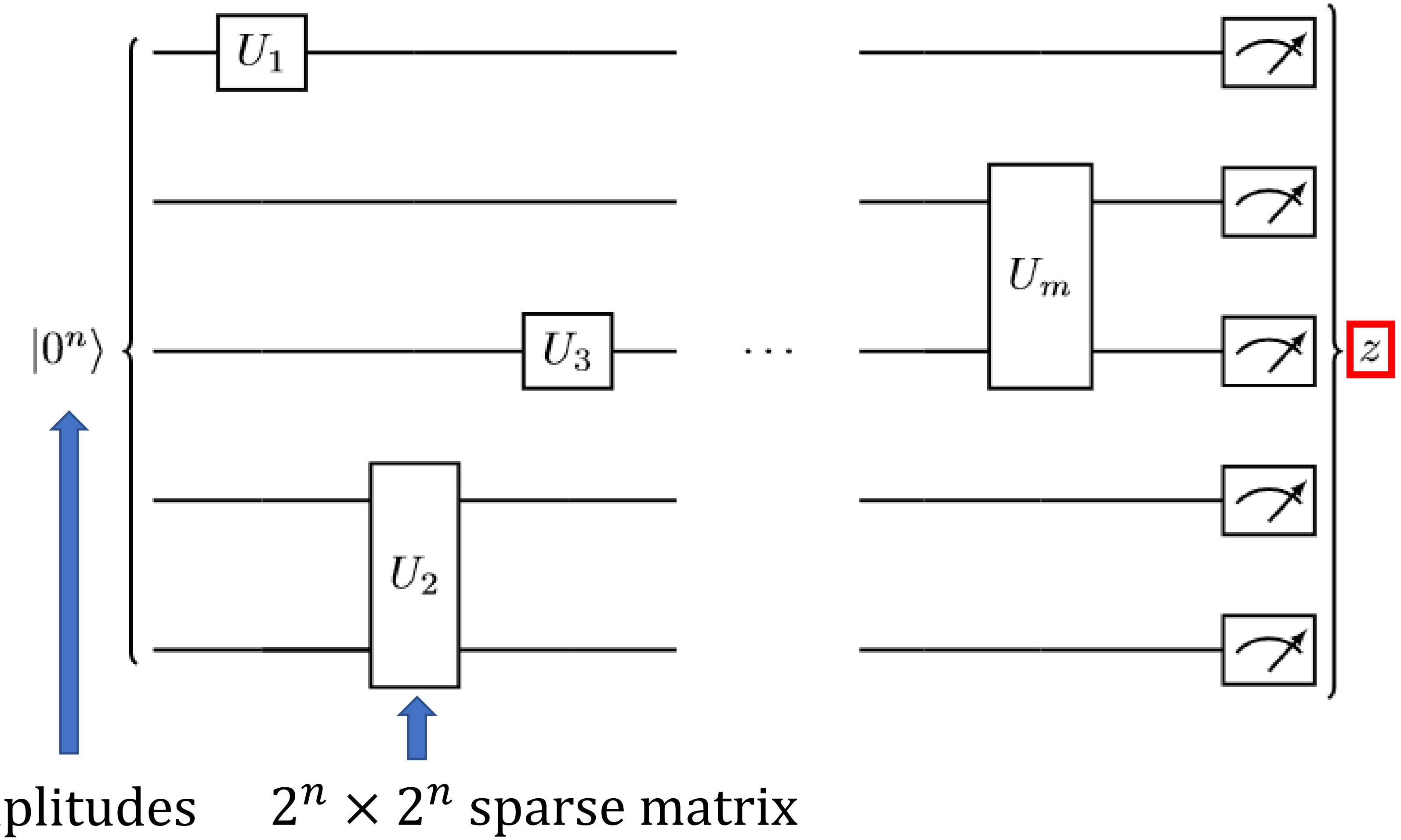
Classical simulation

Task: Sample z



Classical simulation

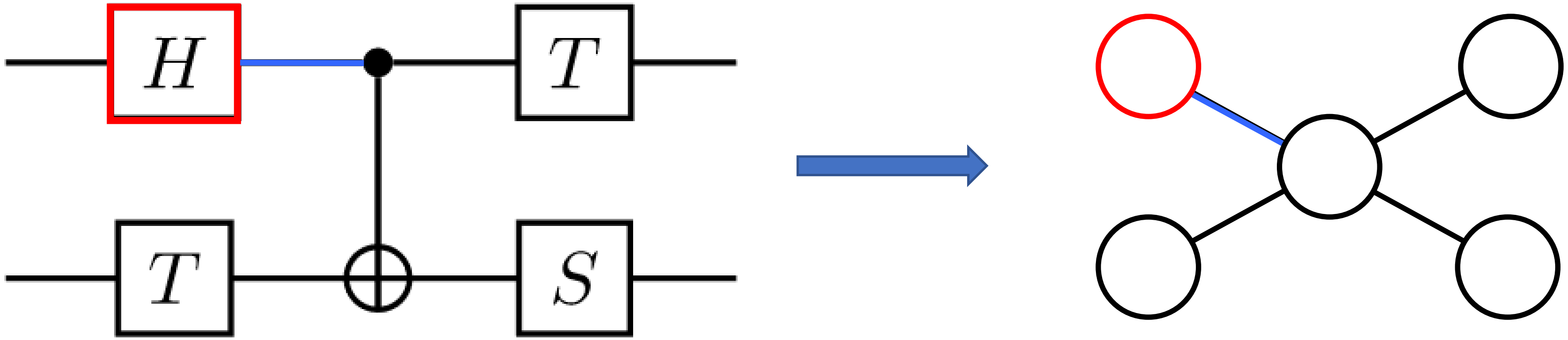
Task: Sample z



Perform m sparse matrix-vector multiplications
Runtime: $2^n m$

Circuit geometry

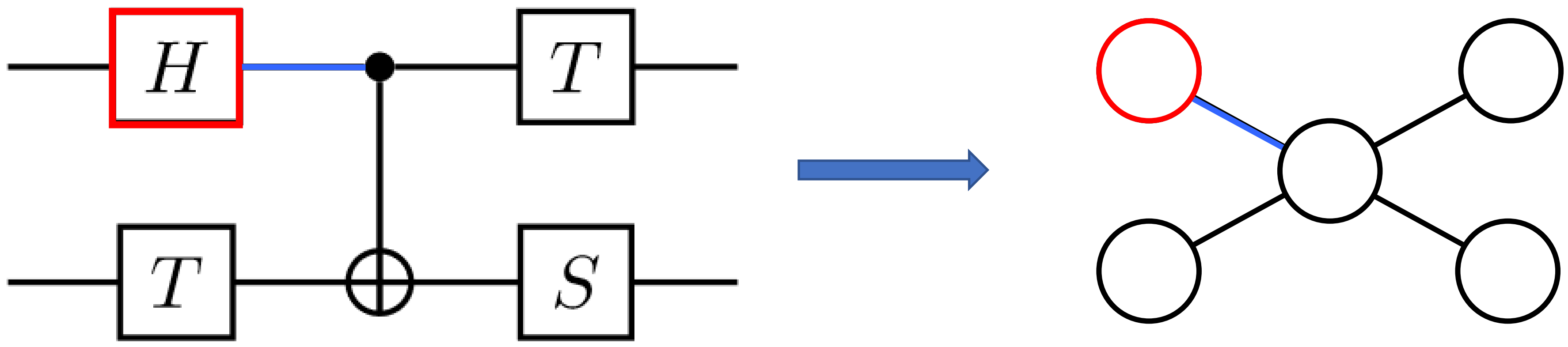
[Markov-Shi 05]



Circuit geometry

[Markov-Shi 05]

Runtime $2^{\text{treewidth}}$ rather than 2^n

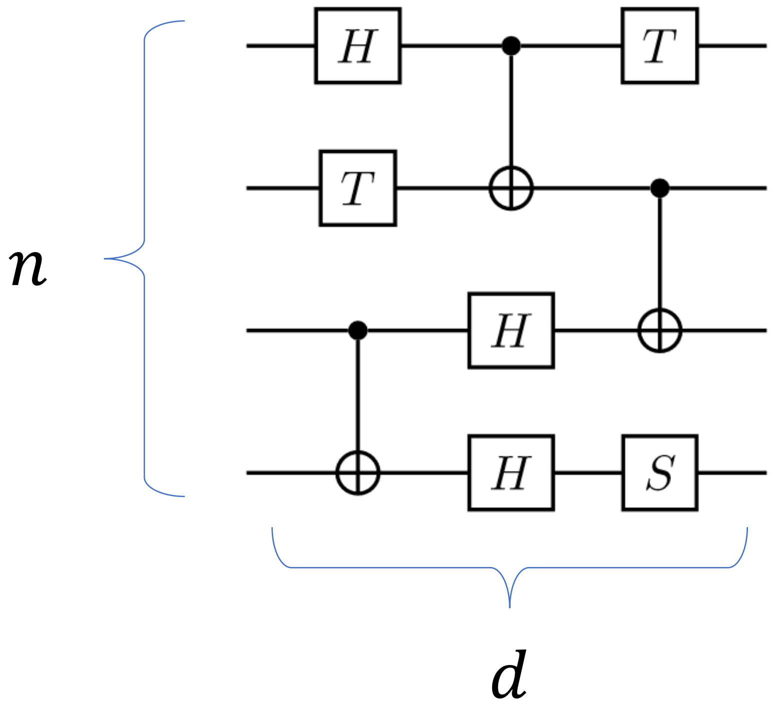


Planar graphs: treewidth= $O\left(\sqrt{|V|}\right)$

Circuit geometry

[Markov-Shi 05]

Runtime $2^{\text{treewidth}}$ rather than 2^n



Constant-depth planar circuits

Runtime: $2^{O(\sqrt{dn})} < 2^n$

Planar graphs: treewidth$= O\left(\sqrt{|V|}\right)$

Clifford circuits

Input $|0^n\rangle$, gate set:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \qquad CZ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Stabilizer formalism
[Gottesman 97],
[Aaronson-Gottesman 04]

Clifford circuits

Input $|0^n\rangle$, gate set:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, \qquad S = \begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix}, \qquad CZ = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & -1\end{bmatrix}$$

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Stabilizer formalism [Gottesman 97], [Aaronson-Gottesman 04]

Graph state formalism [Anders-Briegel 05]

Affine space + phase [Van den Nest 08]

CH form [Bravyi et al 19]

Clifford circuits

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Apply a gate	$O(n)$	[Gottesman 97]
Measure one qubit	$O(n^2)$	[Aaronson-Gottesman 04]
Measure all qubits	$O(n^3)$	[Aaronson-Gottesman 04]

Clifford circuits

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Apply a gate	$O(n)$	[Gottesman 97]
Measure one qubit	$O(n^2)$	[Aaronson-Gottesman 04]
Measure all qubits	$O(n^3)$	[Aaronson-Gottesman 04]
Measure k qubits	$\tilde{O}(n^2 k^{\omega-2})$	[Gosset-Grier-AK-Schaeffer 20]
Measure all qubits	$\tilde{O}(n^{\omega})$	[Gosset-Grier-AK-Schaeffer 20]

$2 \leq \omega < 2.37\ldots$ [Strassen 69],…, [Alman-Williams 19]

Clifford circuits

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Apply a gate	$O(n)$	[Gottesman 97]
Measure one qubit	$O(n^2)$	[Aaronson-Gottesman 04]
Measure all qubits	$O(n^3)$	[Aaronson-Gottesman 04]
Measure k qubits	$\tilde{O}(n^2 k^{\omega-2})$	[Gosset-Grier-AK-Schaeffer 20]
Measure all qubits	$\tilde{O}(n^{\omega})$	[Gosset-Grier-AK-Schaeffer 20]
Any number of CZ gates	$\tilde{O}(n^{\omega})$	[Gosset-Grier-AK-Schaeffer 20]

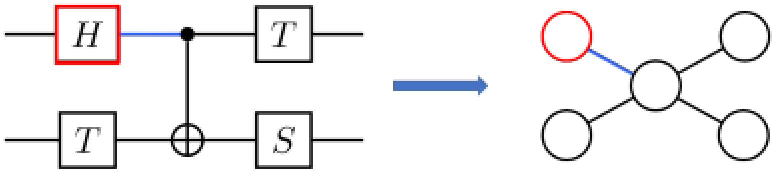
$2 \leq \omega < 2.37\ldots$ [Strassen 69],…, [Alman-Williams 19]

Circuit geometry + Cliffords = ?

Circuit geometry

[Markov-Shi 05]

Runtime $2^{\text{treewidth}}$ rather than 2^n



Planar graphs: treewidth= $O\left(\sqrt{|V|}\right)$

Clifford circuits

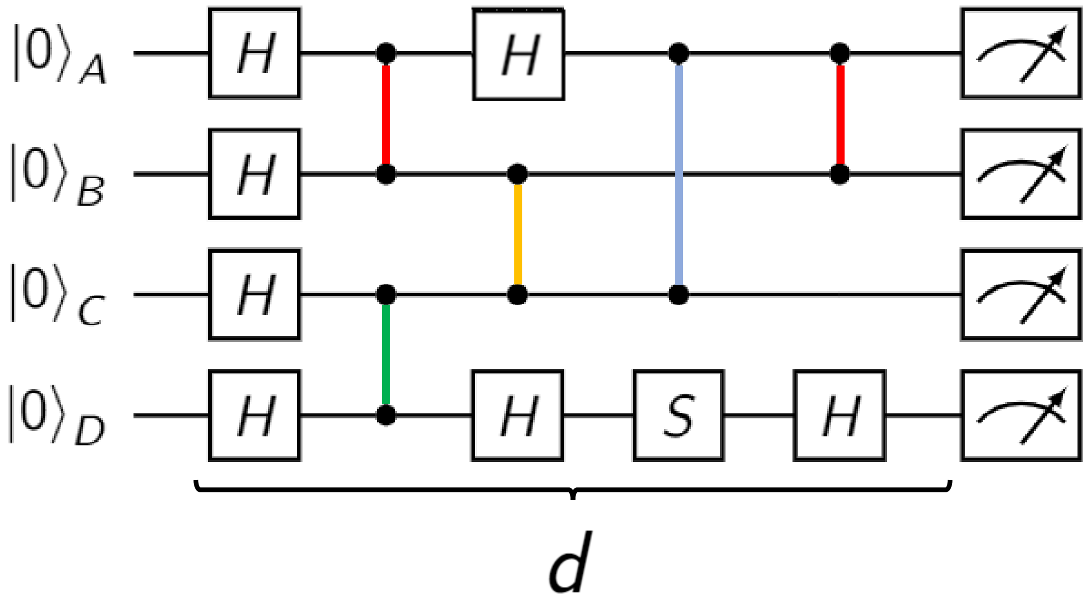
Input $|0^n\rangle$, gate set:

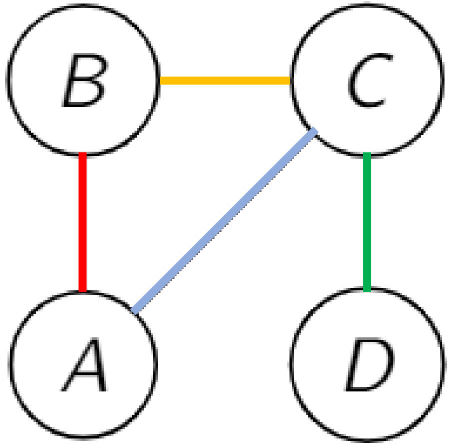
$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \qquad CZ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Gottesman-Knill Theorem [Gottesman 97]
Clifford circuits can be simulated efficiently

Stabilizer formalism
[Gottesman 97],
[Aaronson-Gottesman 04]

Planar Clifford circuits

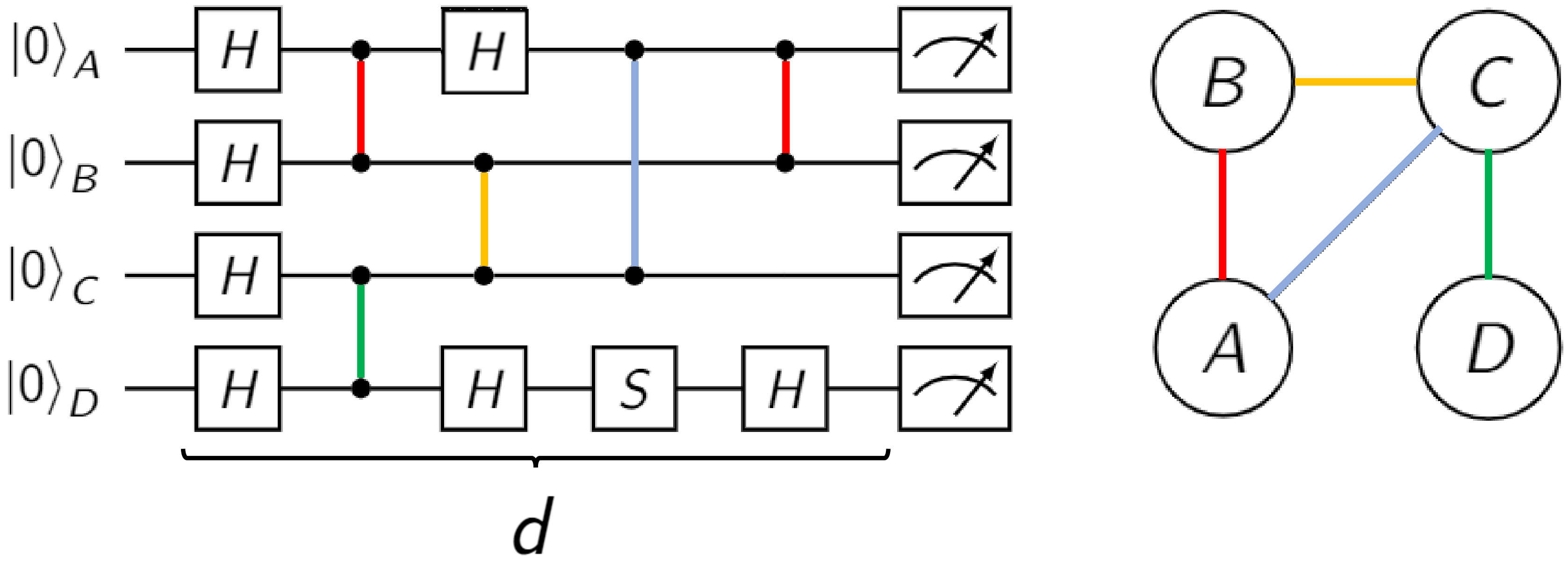




Planar Clifford circuits

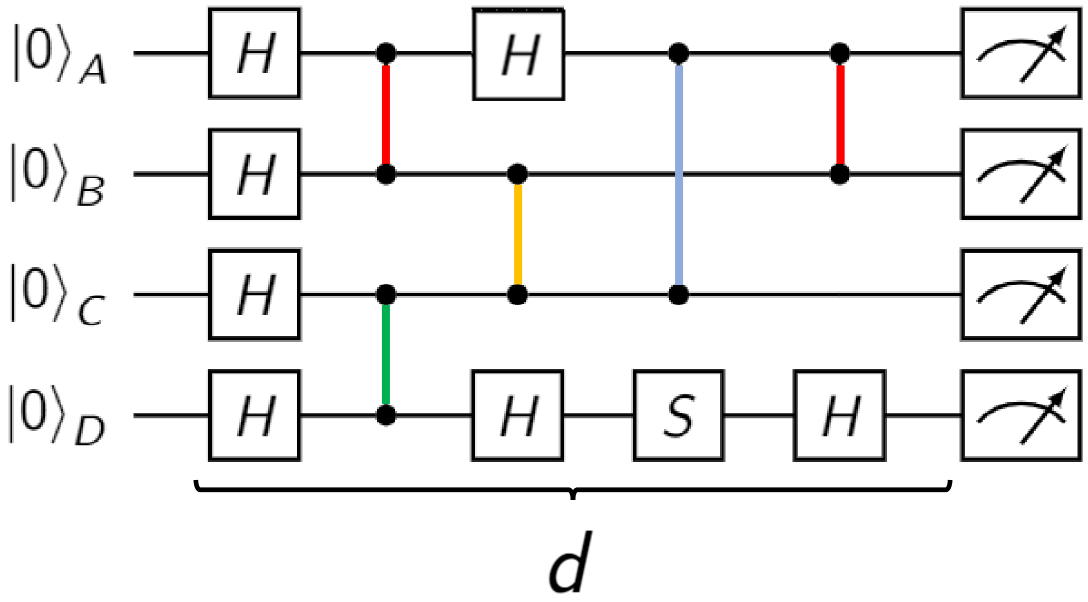
Theorem

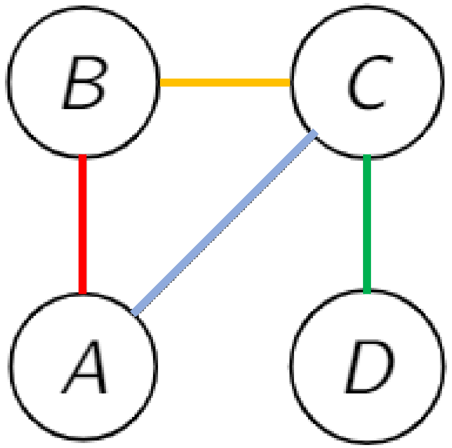
If CZ gates of a Clifford circuit act only on edges of a planar graph, we can sample from the output distribution in time $\tilde{O}(n^{\omega/2}d^{\omega})$



Planar Clifford circuits

Theorem
If CZ gates of a Clifford circuit act only on edges of a planar graph, we can sample from the output distribution in time $\tilde{O}(n^{\omega/2}d^{\omega})$





		Naive		[Aaronson-Gottesman 04]
For $d = O(1)$: $\tilde{O}(n^{\omega/2})$	$<$	$\tilde{O}(n^{\omega})$	$<$	$O(n^3)$

Graph states

Given a graph G,

$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+\rangle^{\otimes n}$$

Graph states

Given a graph G,

$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+\rangle^{\otimes n}$$

Graph state simulation problem

Input: Graph G, Pauli bases $P_v \in \{X, Y, Z\}$ for each v

Task: Sample $z \in \{0,1\}^n$ from

$$\Pr[z] = \left|\langle z | U_{\text{bases}} | G \rangle\right|^2$$

Grid graphs

[Raussendorf-Briegel 01] (MBQC)

Adaptive arbitrary measurements on grid graph states: Universal

Adaptive Pauli measurements on grid graph states: Clifford circuits

Graph state simulation problem

Input: Graph G, Pauli bases $P_v \in \{X, Y, Z\}$ for each v

Task: Sample $z \in \{0,1\}^n$ from

$$\Pr[z] = \left|\langle z|U_{\text{bases}}|G\rangle\right|^2$$

Grid graphs

Graph state simulation problem on $\sqrt{n} \times \sqrt{n}$ grid graph

[Bravyi-Gosset-König 18]

Quantum algorithm: constant-depth

Classical algorithm: must have at least log-depth

Grid graphs

Graph state simulation problem on $\sqrt{n} \times \sqrt{n}$ grid graph

[Bravyi-Gosset-König 18]

Quantum algorithm: constant-depth

Classical algorithm: must have at least log-depth

Goal: Study gate complexity

Gate complexity of graph state simulation

Quantum

Gate complexity $= O(|E| + |V|)$

$O(n)$ for planar graphs

$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+\rangle^{\otimes n}$$

Gate complexity of graph state simulation

Quantum

Gate complexity $= O(|E| + |V|)$

$O(n)$ for planar graphs

$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+\rangle^{\otimes n}$$

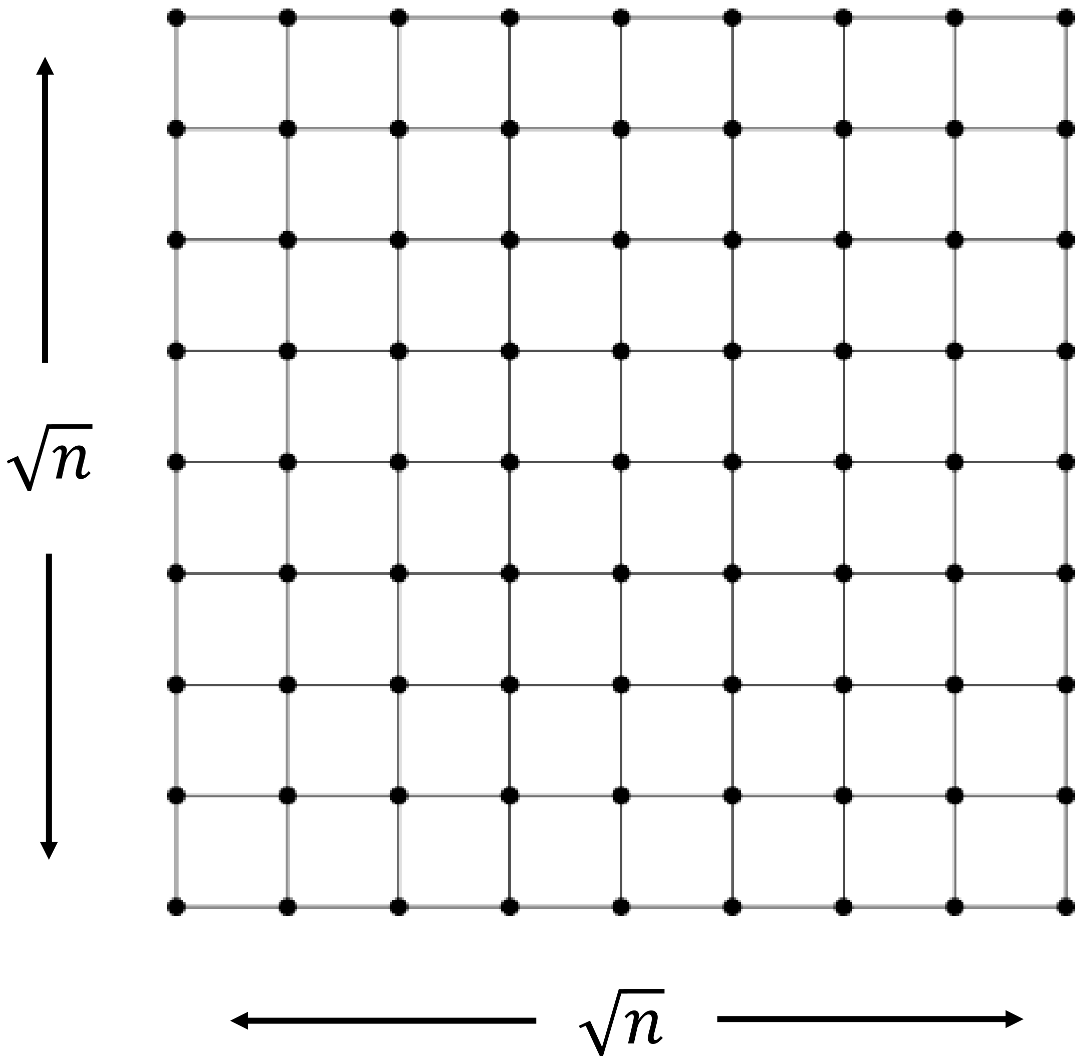
Theorem

The graph state simulation problem can be solved classically in time $\tilde{O}\left(n^{\omega/2}\right)$ for planar graphs

	Naive	[Aaronson-Gottesman 04]
$\tilde{O}\left(n^{\omega/2}\right)$ $<$	$\tilde{O}(n^{\omega})$ $<$	$O(n^3)$

Warmup

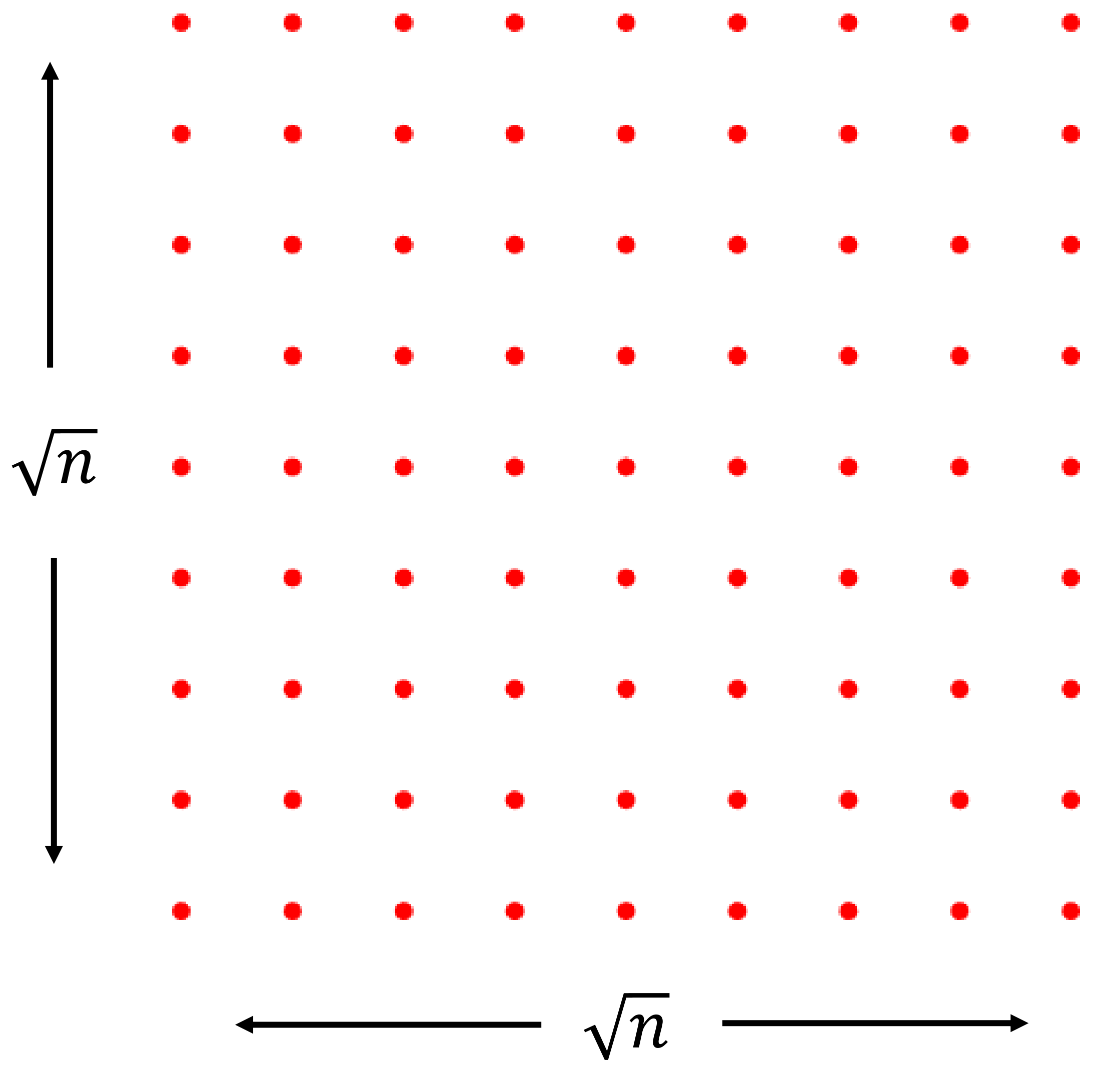
Naïve approach



Warmup

<u>Naïve approach</u>

Initialize n qubits $\qquad O(n^{\omega})$

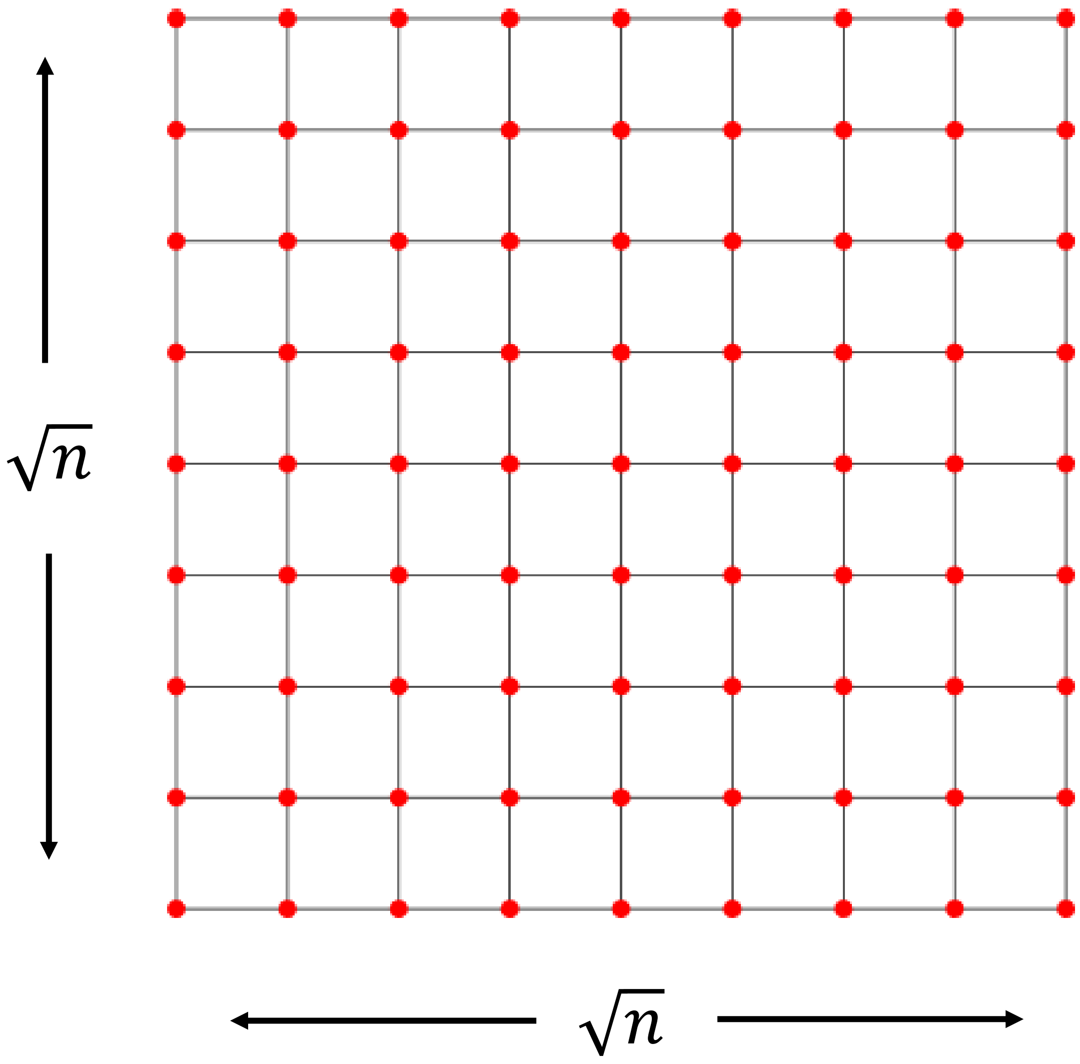


Warmup

<u>Naïve approach</u>

Initialize n qubits $\tilde{O}(n^{\omega})$

Apply CZ gates $\tilde{O}\ (n^{\omega})$



Warmup

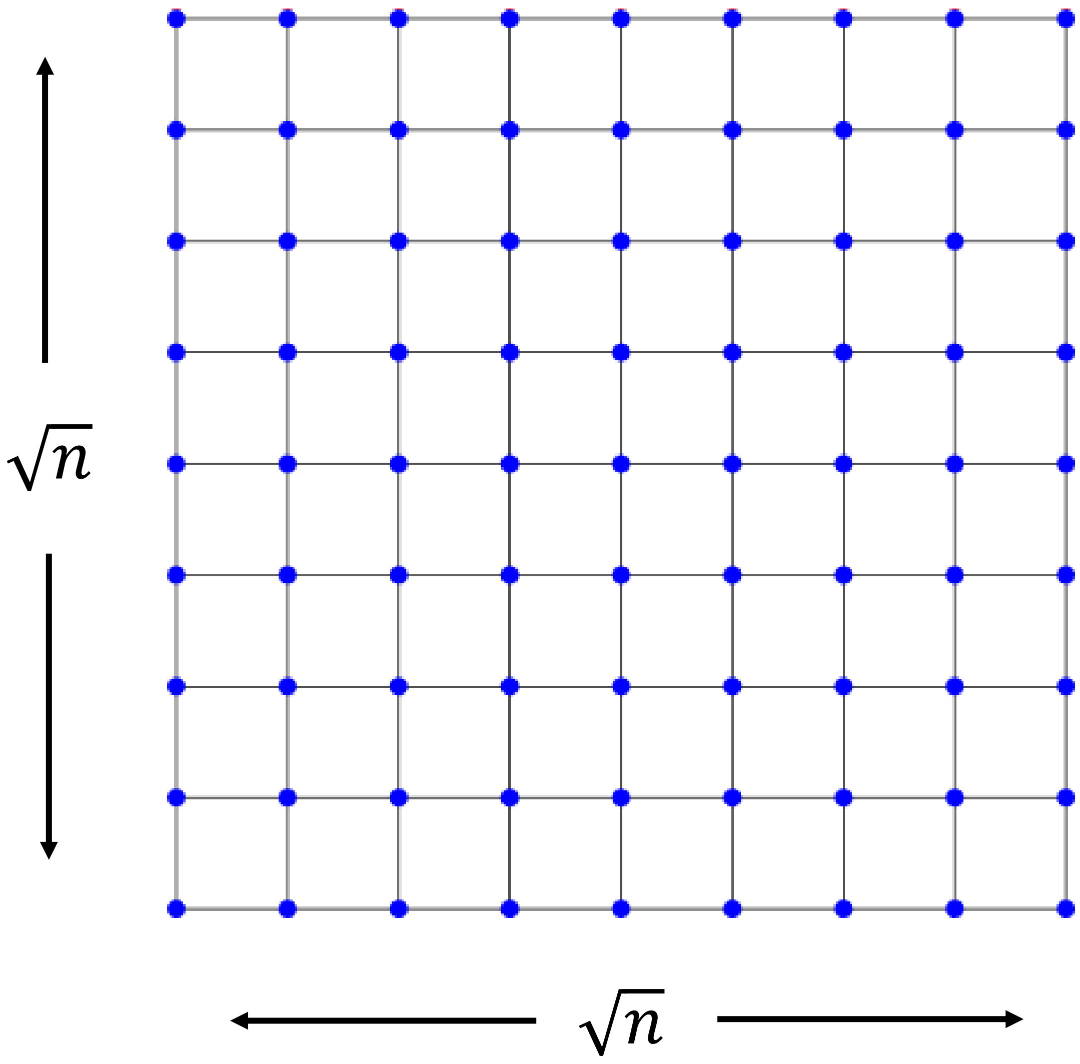
<u>Naïve approach</u>

Initialize n qubits $\tilde{O}(n^\omega)$

Apply CZ gates $\tilde{O}(n^\omega)$

Measure all qubits $\tilde{O}(n^\omega)$

Total runtime: $\tilde{O}(n^\omega)$



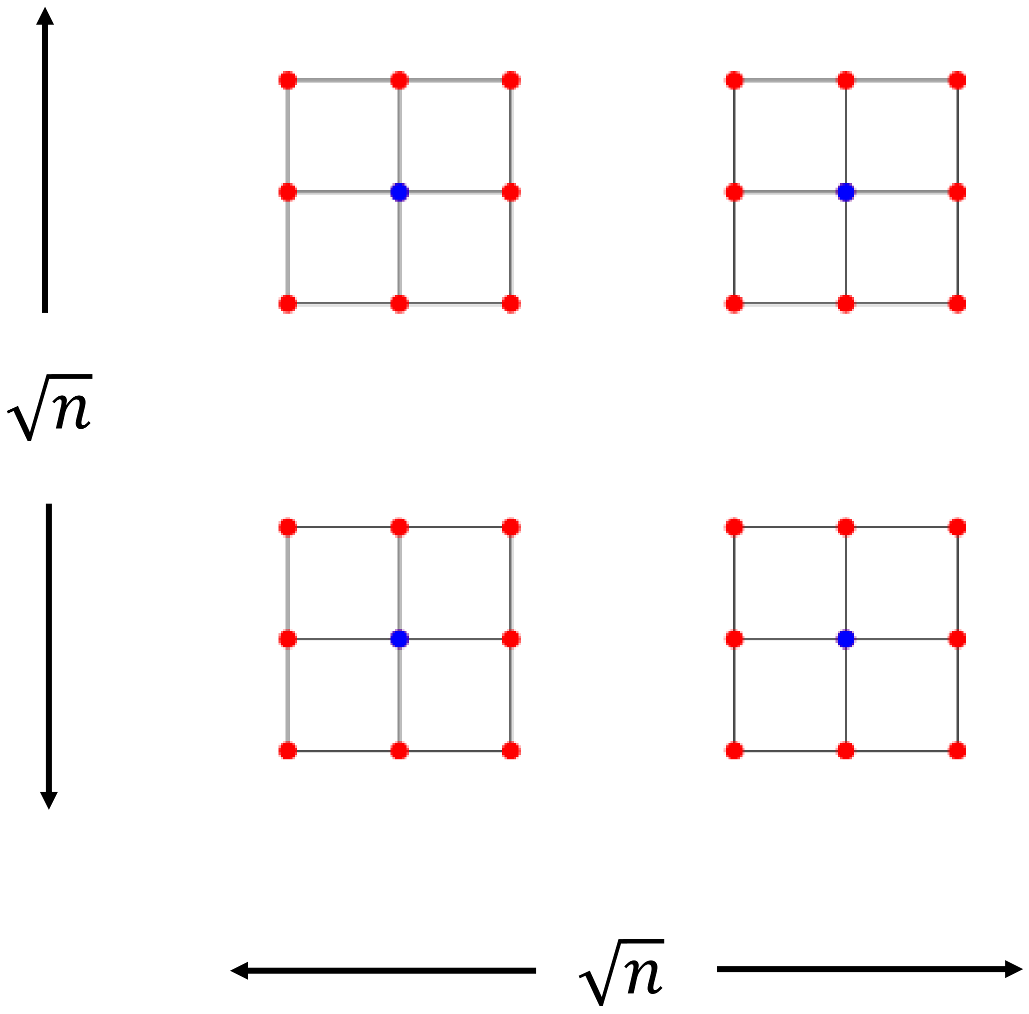
Warmup

● $= O(\sqrt{n})$

Improved approach

Recursion on subgrids

$4 \cdot T(n/4)$



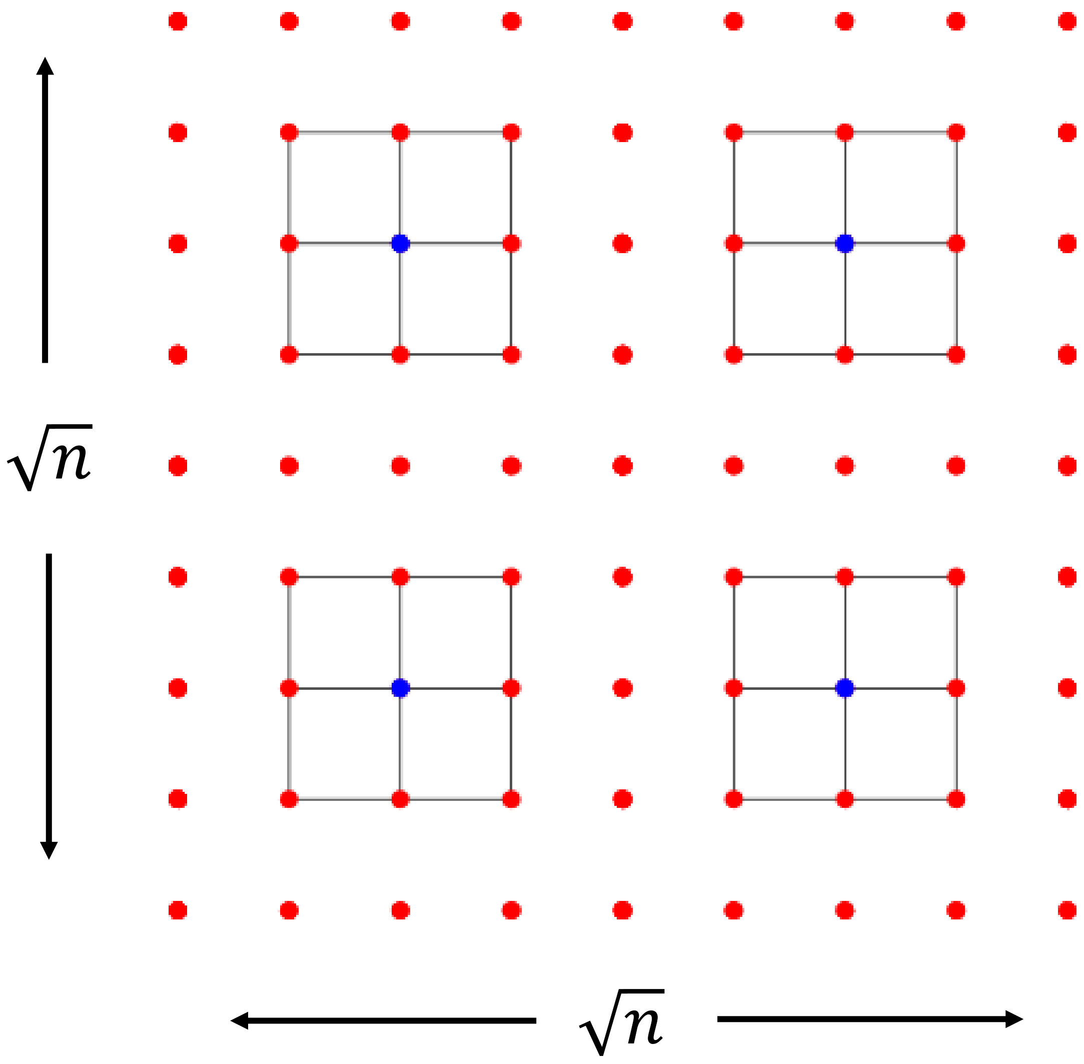
Warmup

(red dot) $= O(\sqrt{n})$

<u>Improved approach</u>

Recursion on subgrids $4 \cdot T(n/4)$

Initialize remaining qubits $\tilde{O}(n^{\omega/2})$

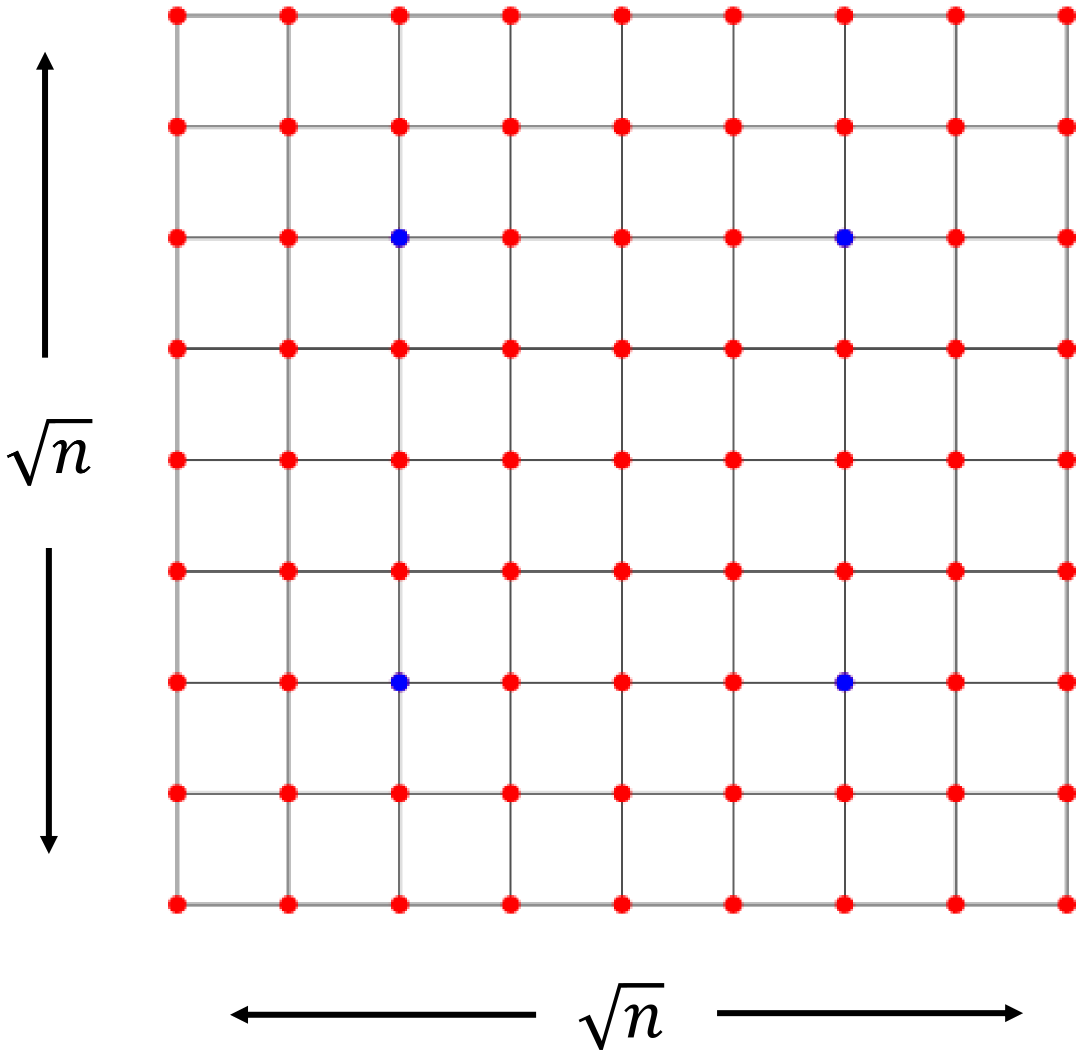


Warmup

● = $O(\sqrt{n})$

<u>Improved approach</u>

Recursion on subgrids	$4 \cdot T(n/4)$
Initialize remaining qubits	$\tilde{O}\left(n^{\omega/2}\right)$
Apply gates	$\tilde{O}\left(n^{\omega/2}\right)$

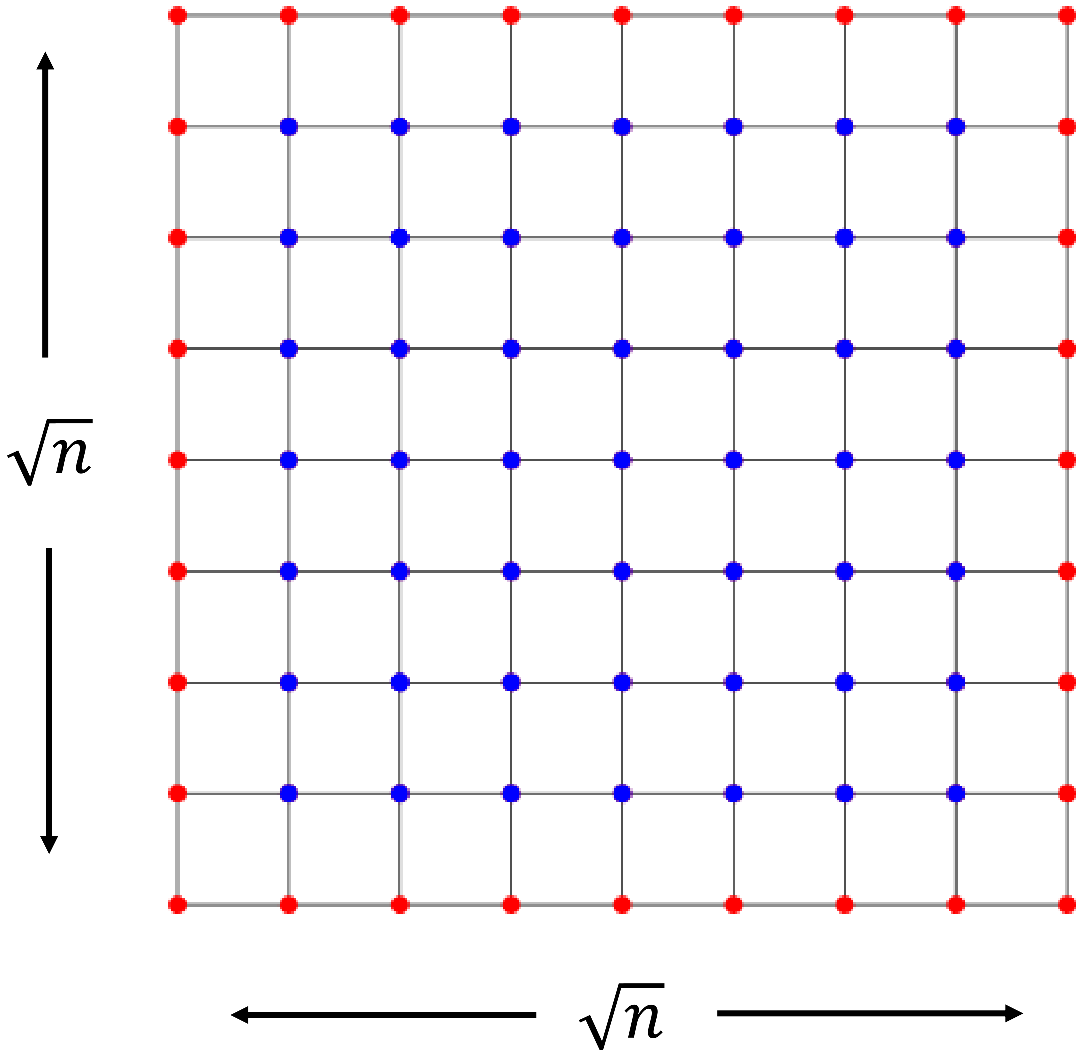


Warmup

$\bullet = O(\sqrt{n})$

<u>Improved approach</u>

Recursion on subgrids	$4 \cdot T(n/4)$
Initialize remaining qubits	$\tilde{O}\left(n^{\omega/2}\right)$
Apply gates	$\tilde{O}\left(n^{\omega/2}\right)$
Measure	$\tilde{O}\left(n^{\omega/2}\right)$



Warmup

<u>Improved approach</u>

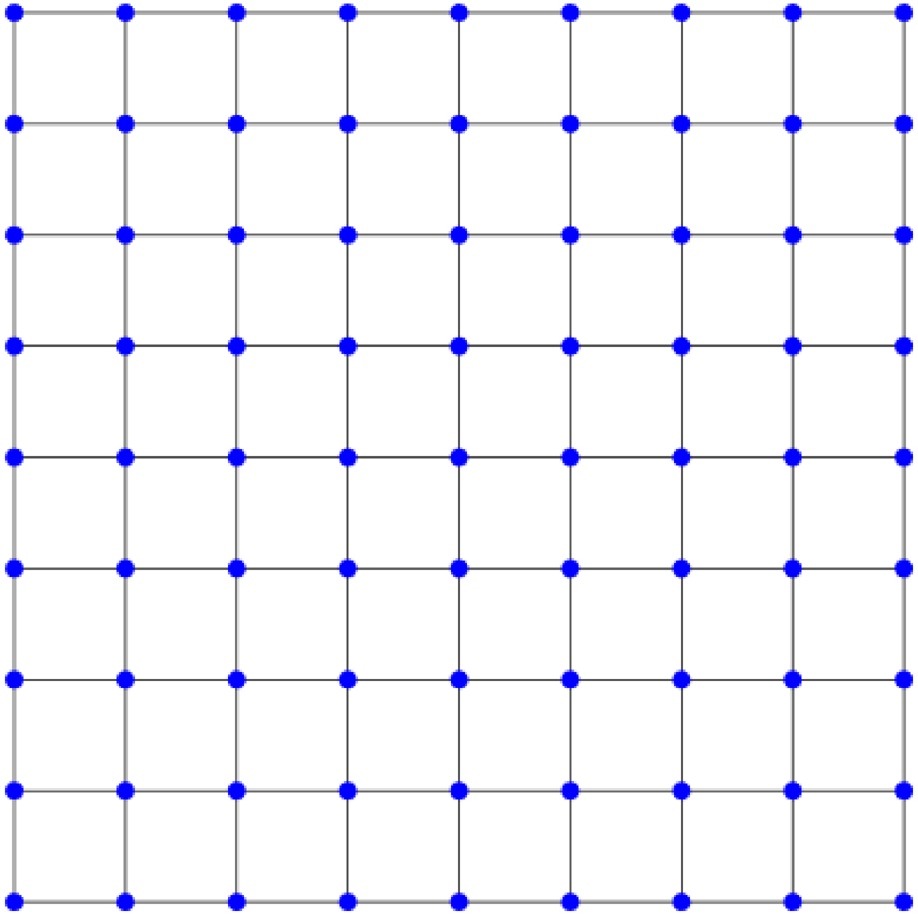
Recursion on subgrids $4 \cdot T(n/4)$

Initialize remaining qubits $\tilde{O}\left(n^{\omega/2}\right)$

Apply gates $\tilde{O}\left(n^{\omega/2}\right)$

Measure $\tilde{O}\left(n^{\omega/2}\right)$

Total runtime: $4 \cdot T(n/4) + \tilde{O}\left(n^{\omega/2}\right) = \tilde{O}\left(n^{\omega/2}\right)$



<u>Key idea:</u> Schedule operations to minimize qubits stored in memory

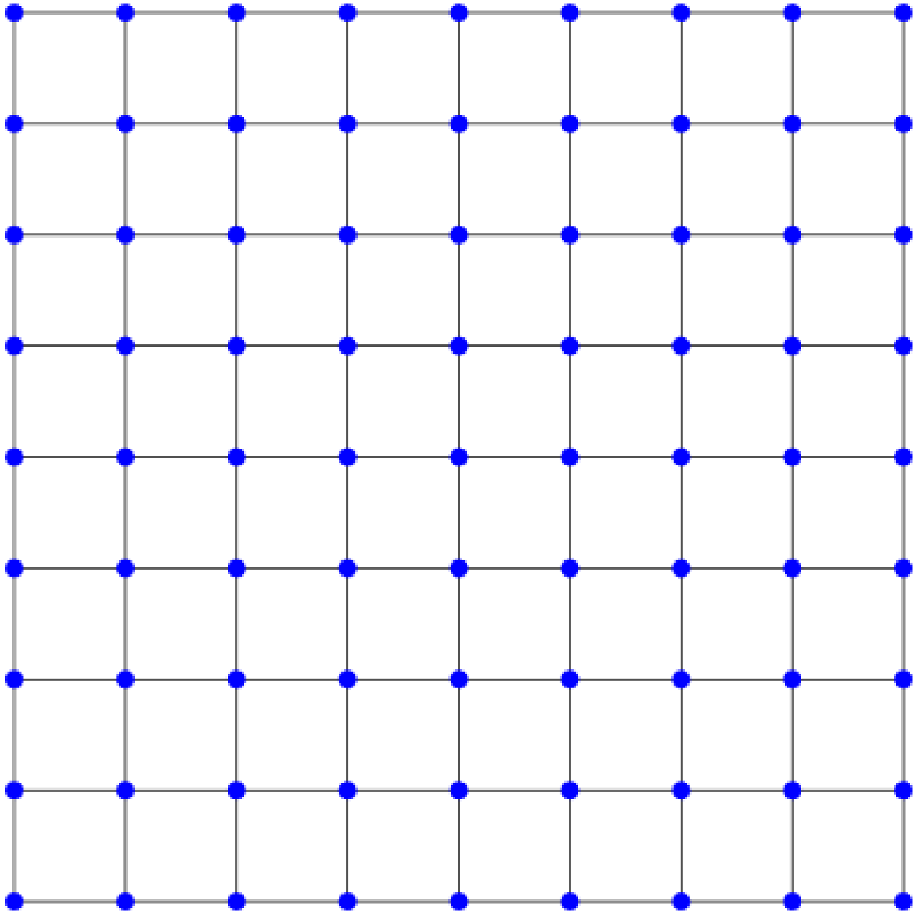
Warmup

Nested dissection
[George 73], [Lipton-Rose-Tarjan 79], [Alon-Yuster 10]

<u>Improved approach</u>

Recursion on subgrids	$4 \cdot T(n/4)$
Initialize remaining qubits	$\tilde{O}\left(n^{\omega/2}\right)$
Apply gates	$\tilde{O}\left(n^{\omega/2}\right)$
Measure	$\tilde{O}\left(n^{\omega/2}\right)$

Total runtime: $4 \cdot T(n/4) + \tilde{O}\left(n^{\omega/2}\right) = \tilde{O}\left(n^{\omega/2}\right)$

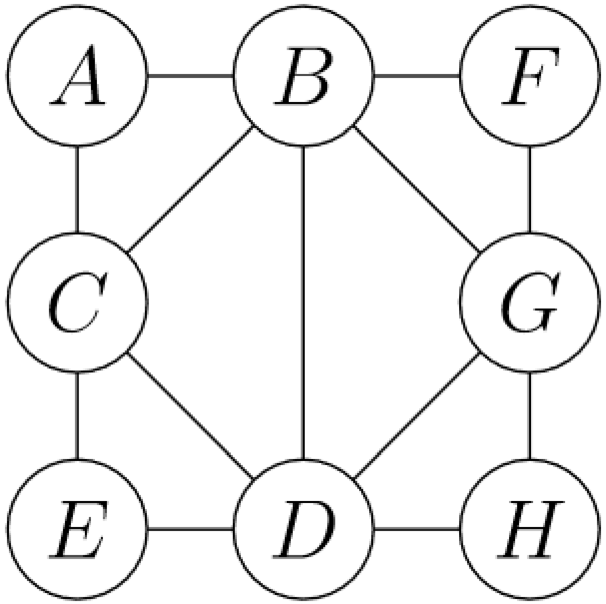


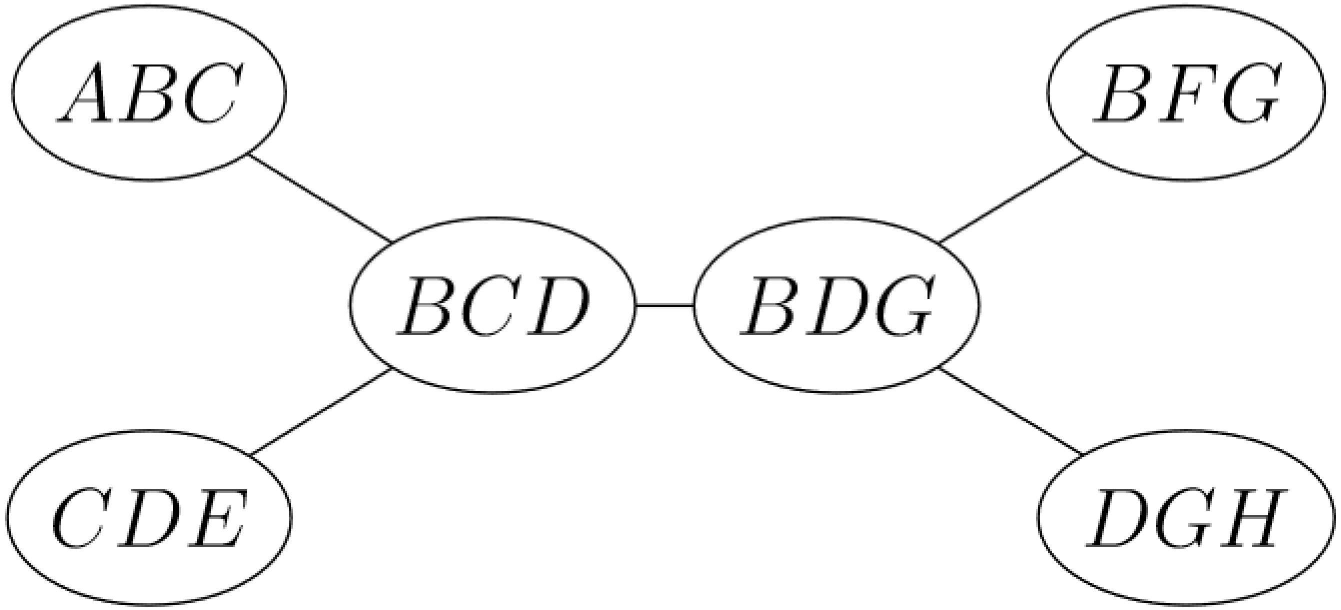
<u>Key idea:</u> Schedule operations to minimize qubits stored in memory

Tree decompositions

Set of *bags* $B_i \subseteq V(G)$ arranged in a tree such that

- Each vertex appears somewhere
- Each edge appears somewhere
- Bags containing a given vertex form a connected subtree

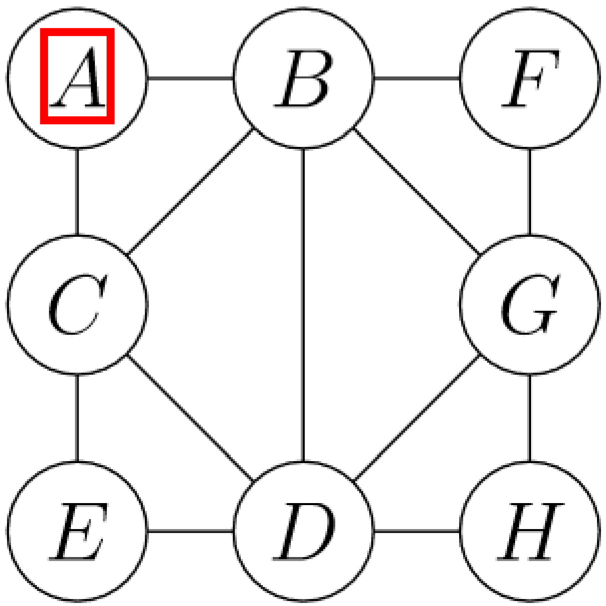


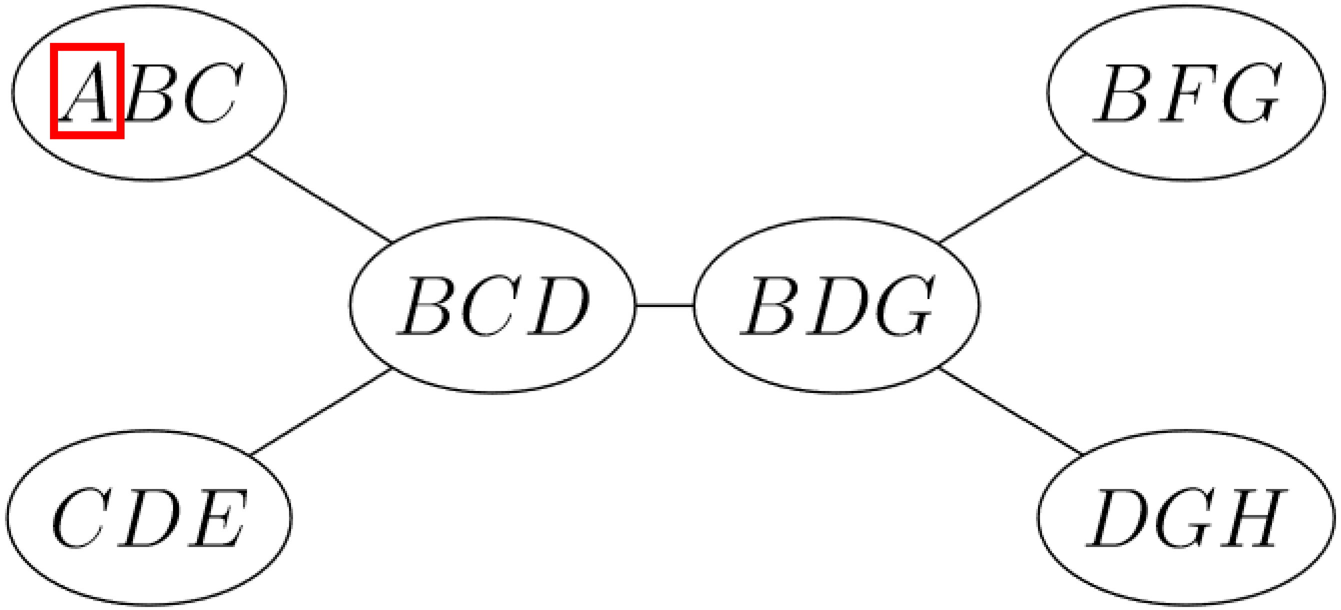


Tree decompositions

Set of *bags* $B_i \subseteq V(G)$ arranged in a tree such that

- Each vertex appears somewhere
- Each edge appears somewhere
- Bags containing a given vertex form a connected subtree

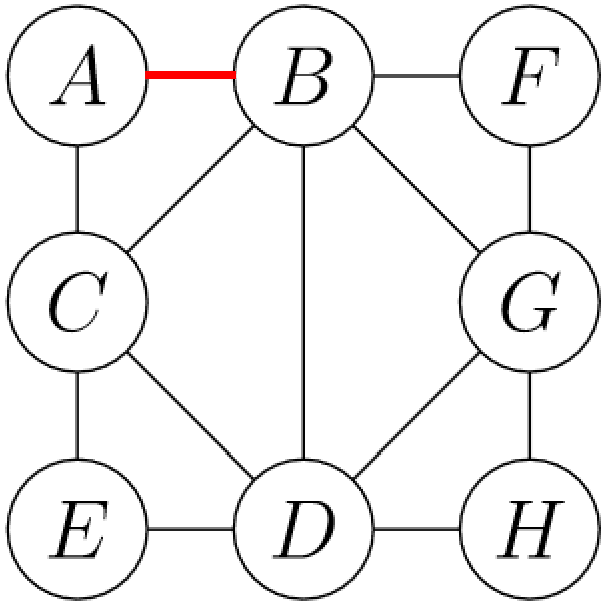


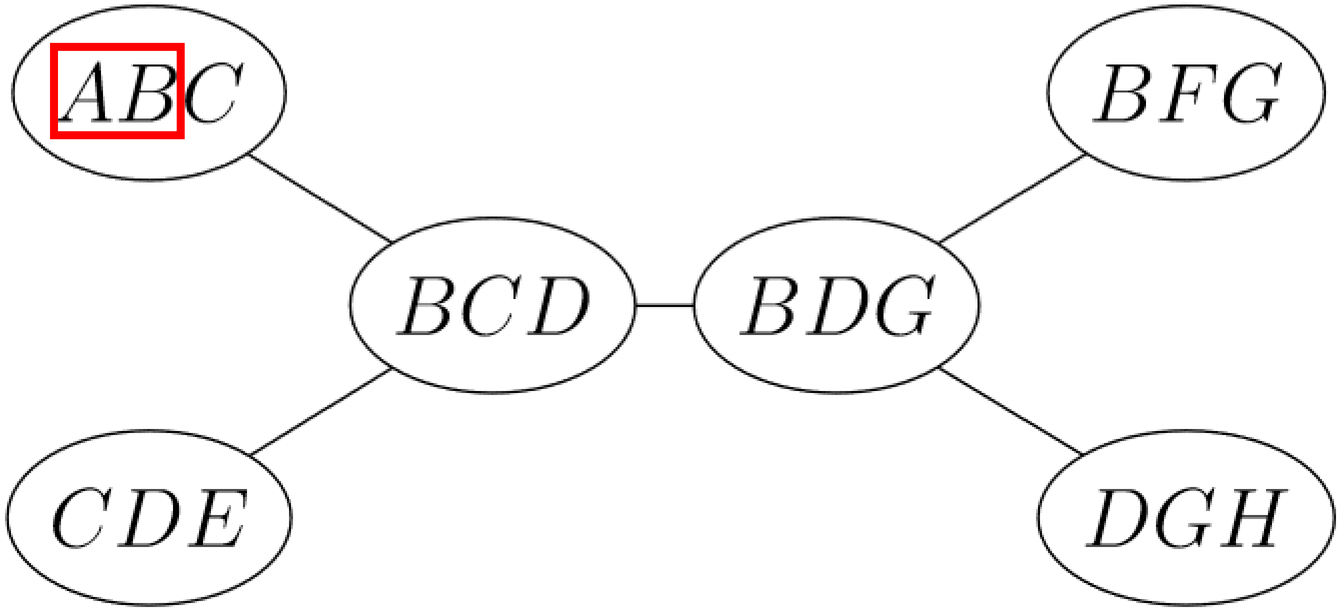


Tree decompositions

Set of *bags* $B_i \subseteq V(G)$ arranged in a tree such that

- Each vertex appears somewhere
- Each edge appears somewhere
- Bags containing a given vertex form a connected subtree

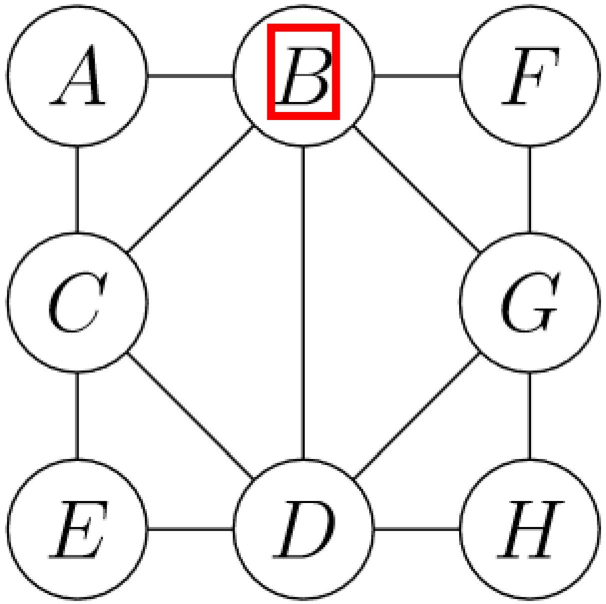


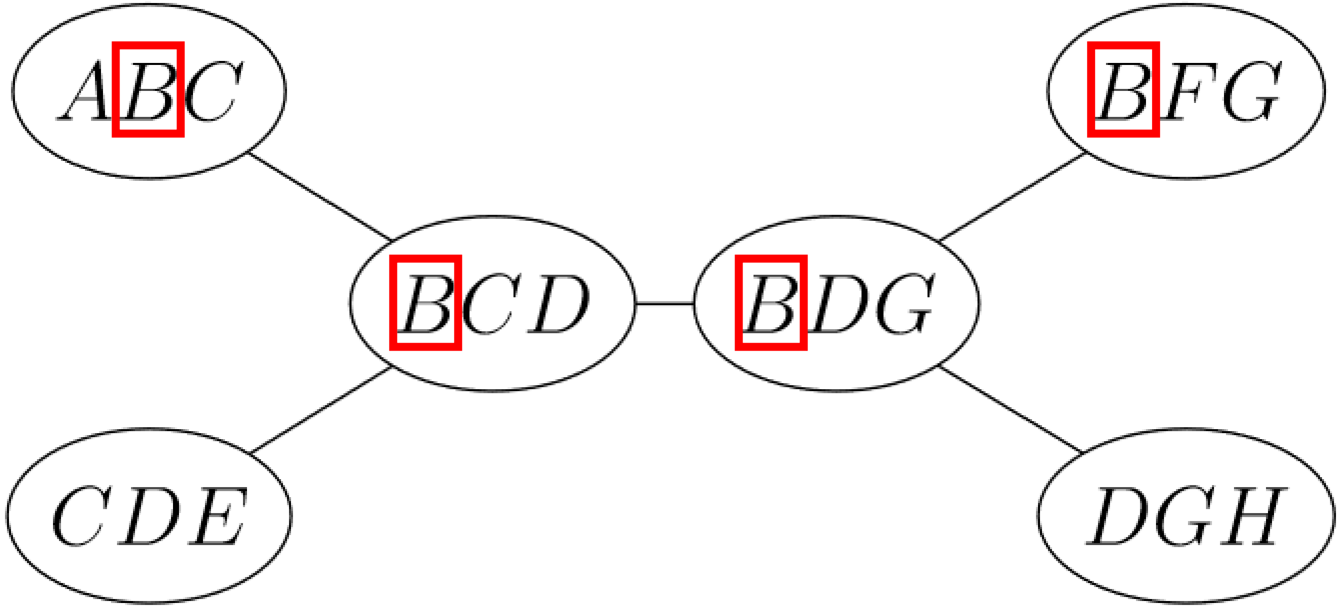


Tree decompositions

Set of *bags* $B_i \subseteq V(G)$ arranged in a tree such that

- Each vertex appears somewhere
- Each edge appears somewhere
- Bags containing a given vertex form a connected subtree





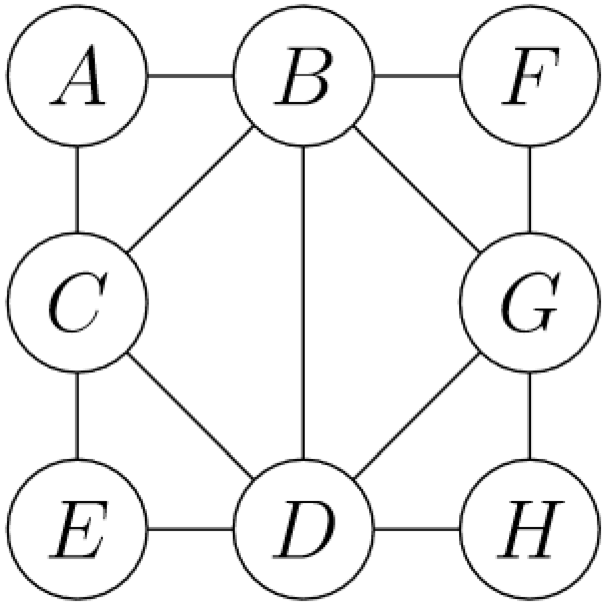
Tree decompositions

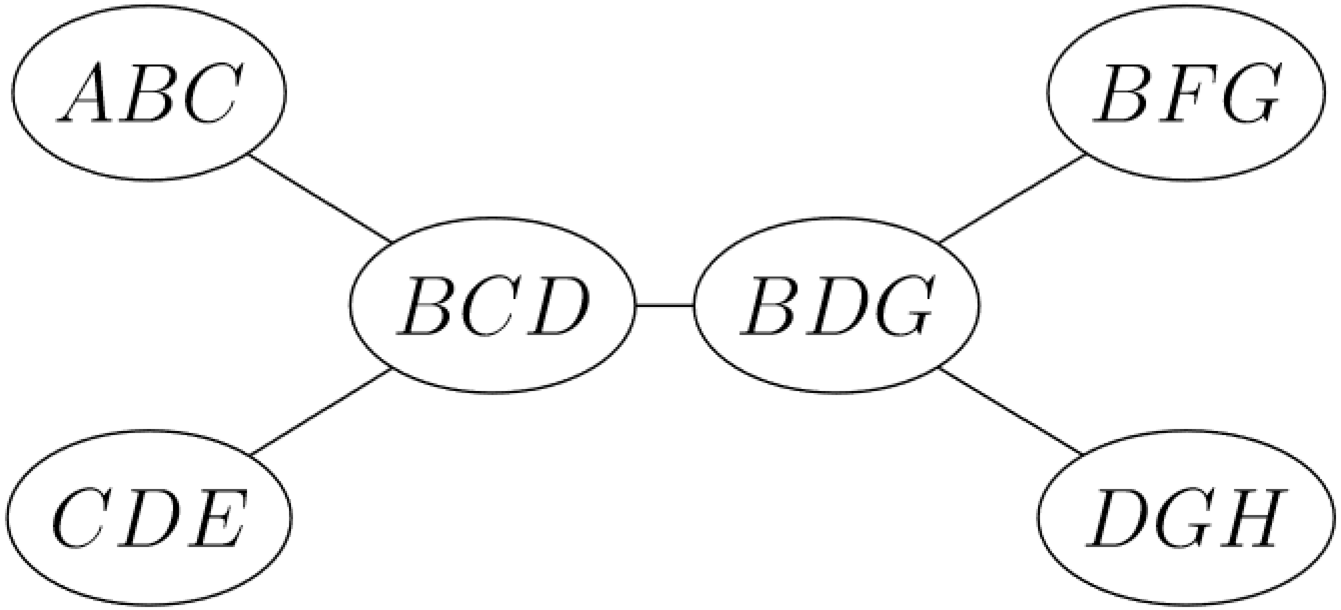
Width of decomposition T

$$|T| = \max_i |B_i| - 1$$

Treewidth of graph

$$\mathrm{t}(G) = \min |T|$$





Tree decompositions

Width of decomposition T

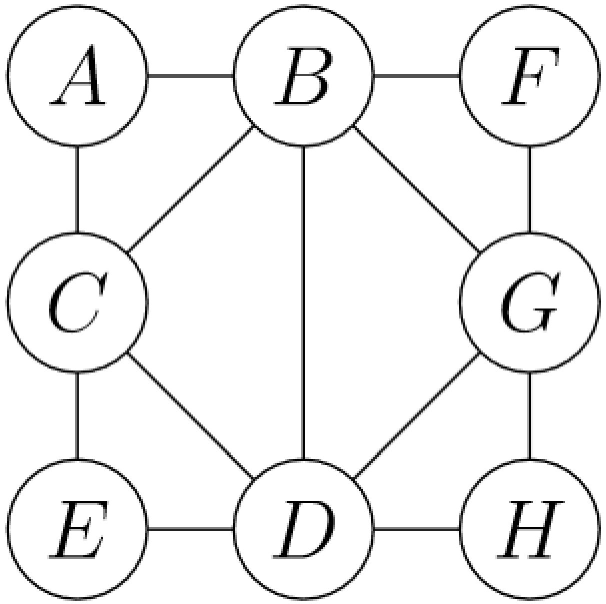
$$|T| = \max_i |B_i| - 1$$

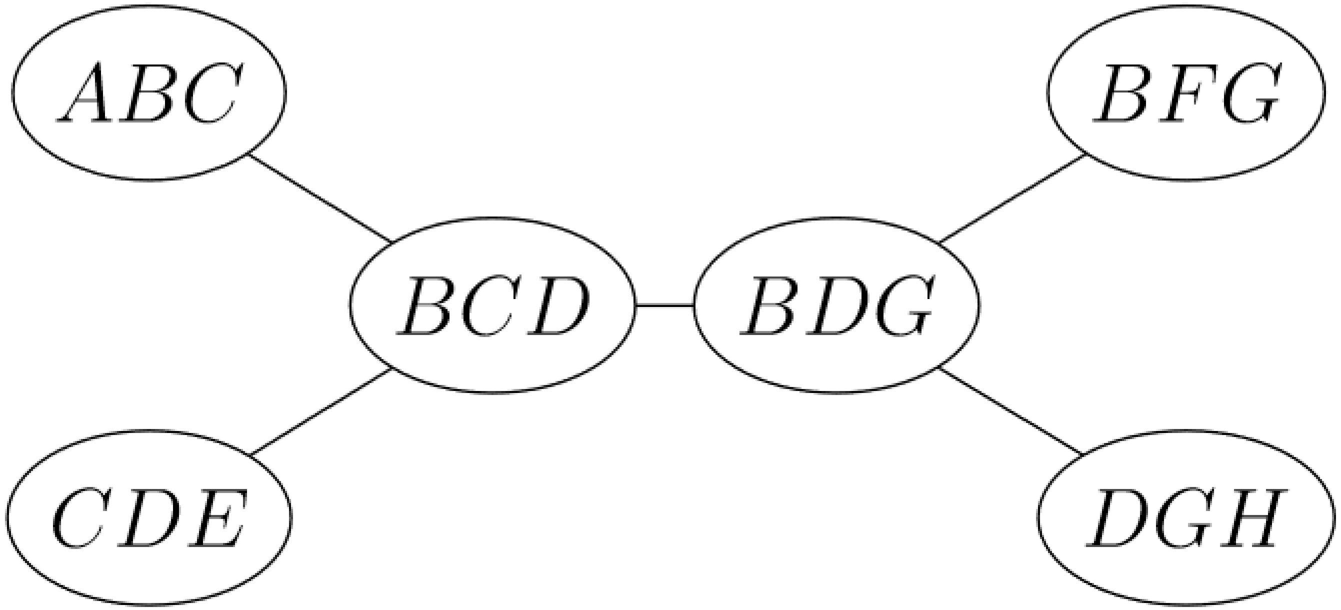
Treewidth of graph

$$\mathrm{t}(G) = \min |T|$$

Planar graphs

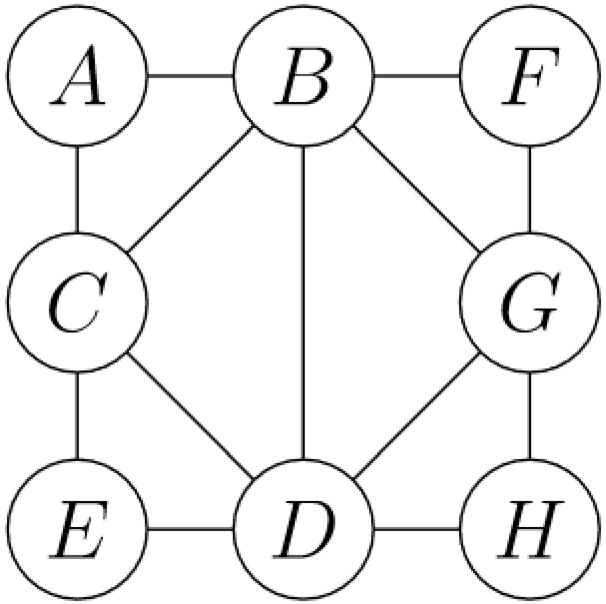
$$t(G) = O\left(\sqrt{|V|}\right)$$

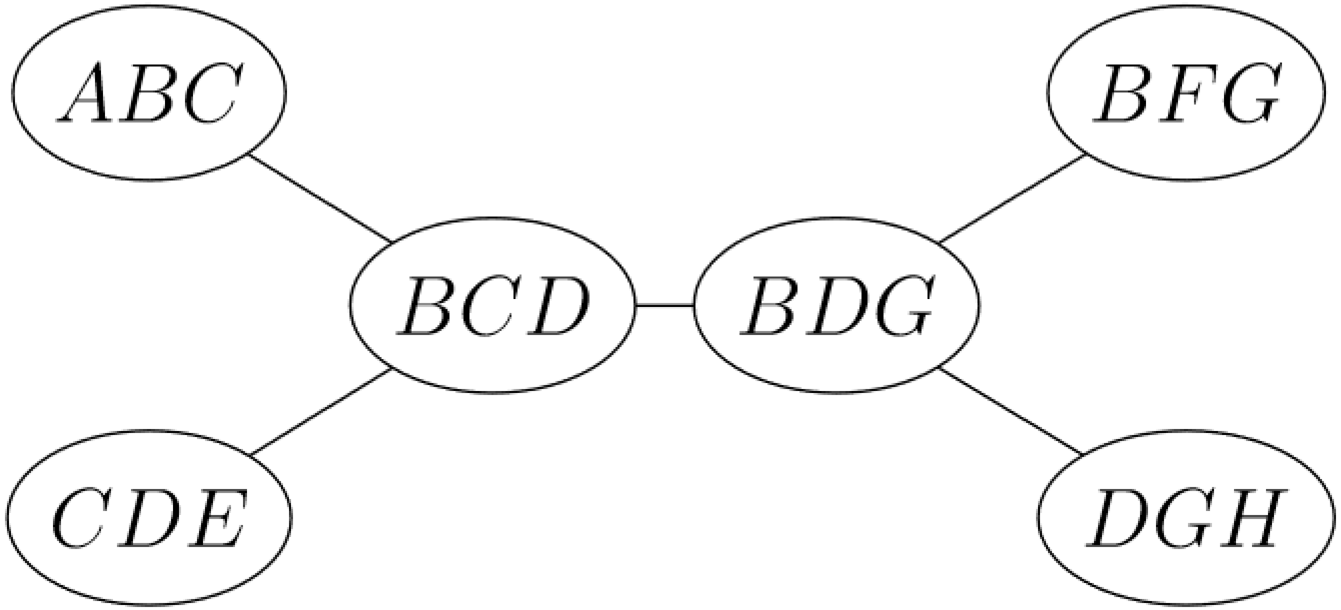




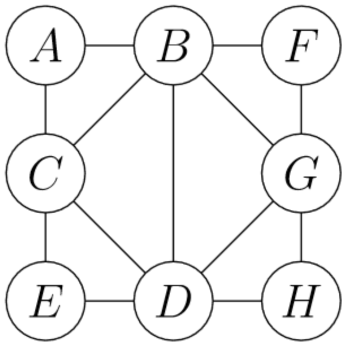
Tree decompositions

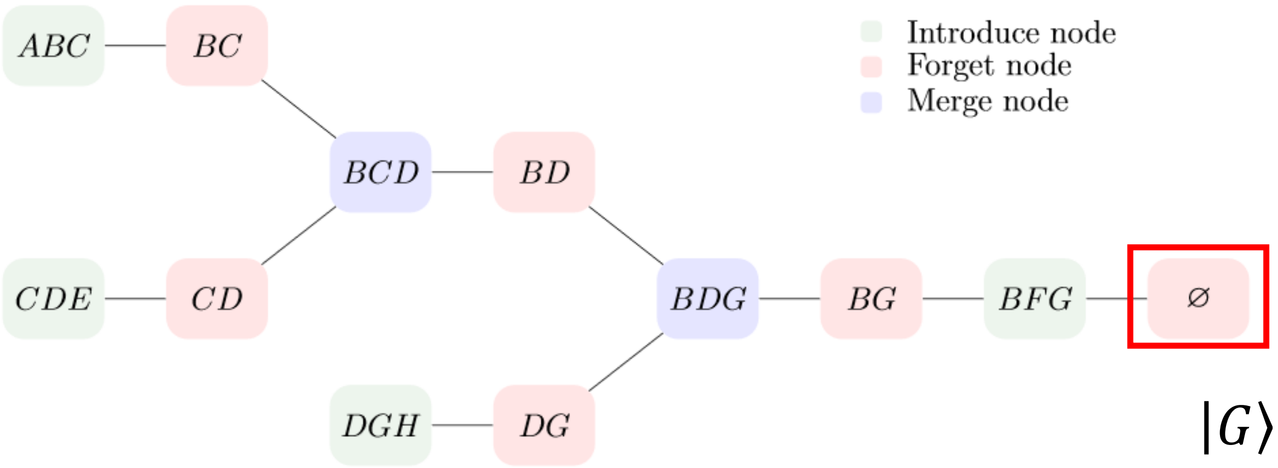
Idea: Use tree decomposition to derive schedule



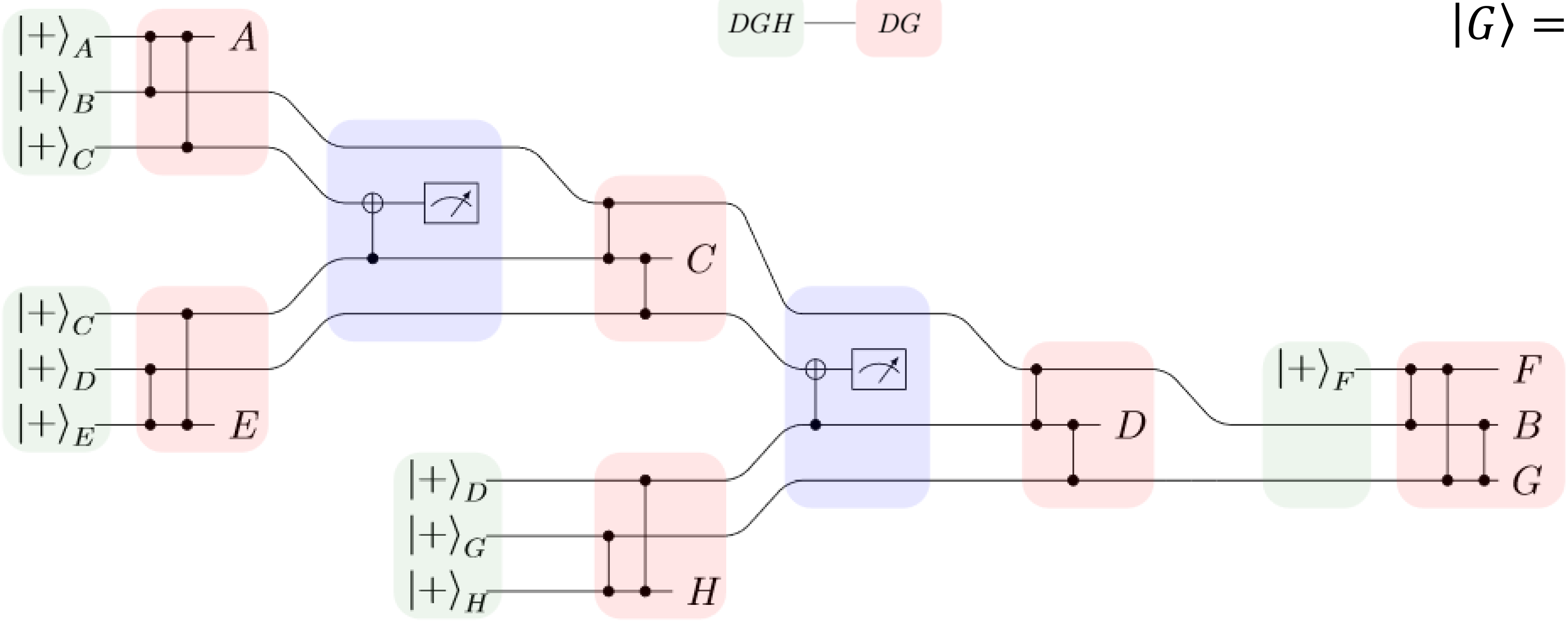


Tree decomposition ↦ circuit

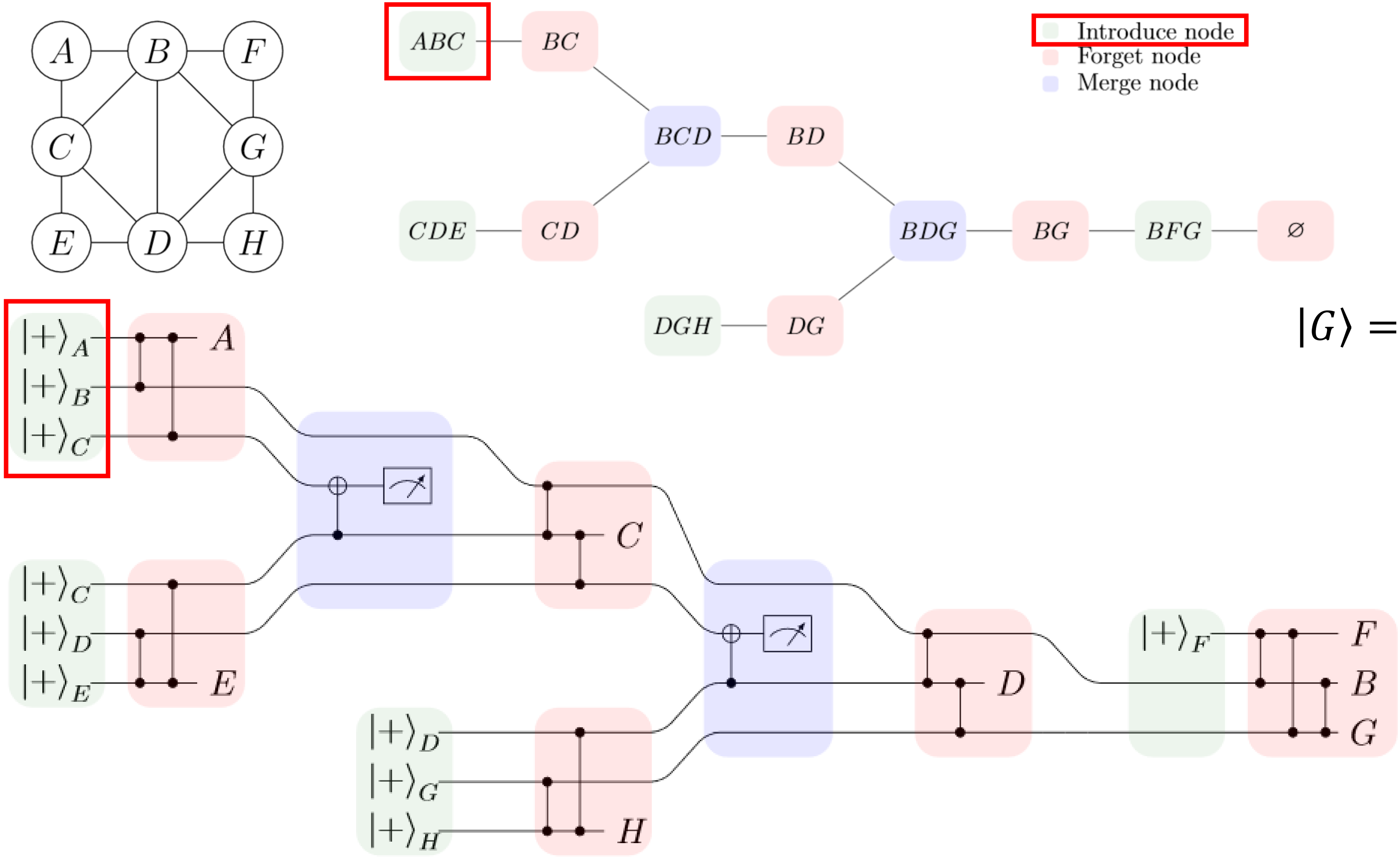




$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+^{\otimes n}\rangle$$

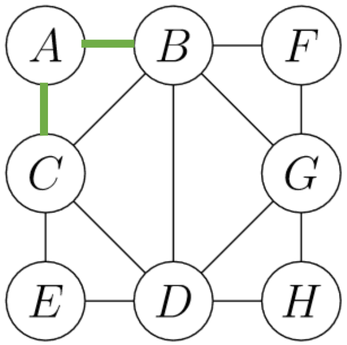


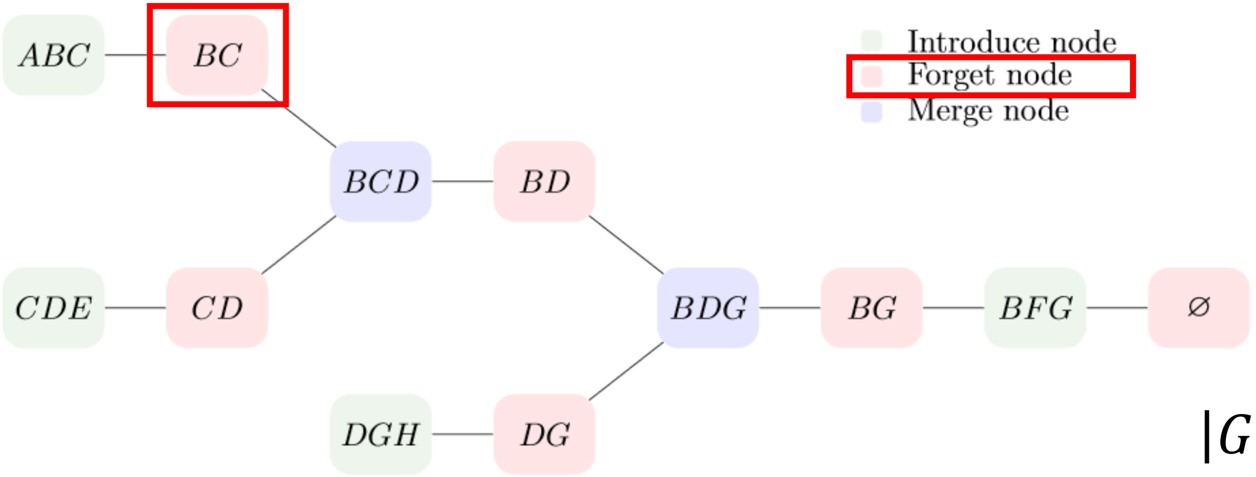
Tree decomposition ↦ circuit



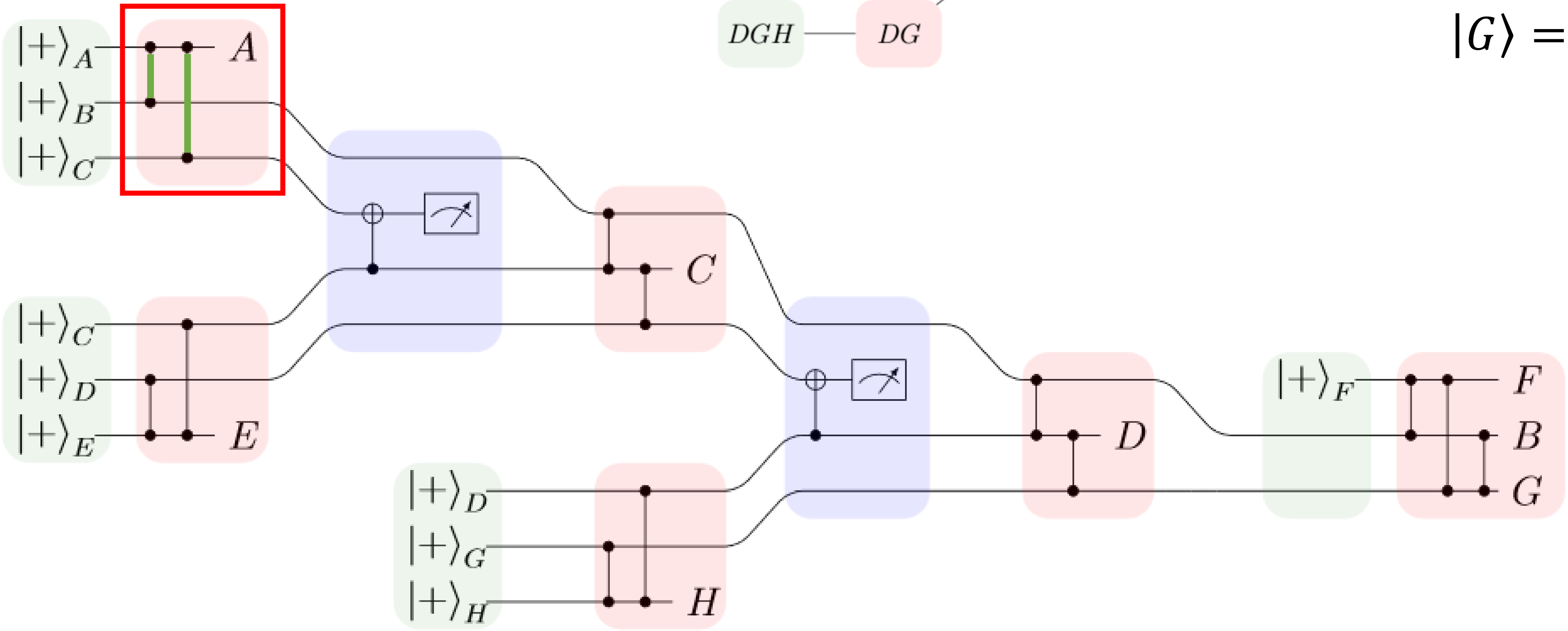
$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+^{\otimes n}\rangle$$

Tree decomposition ↦ circuit

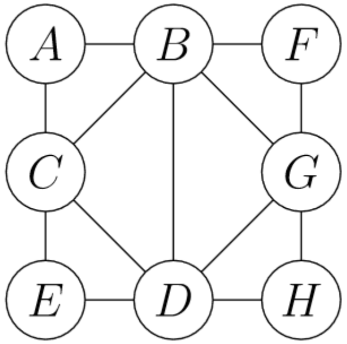


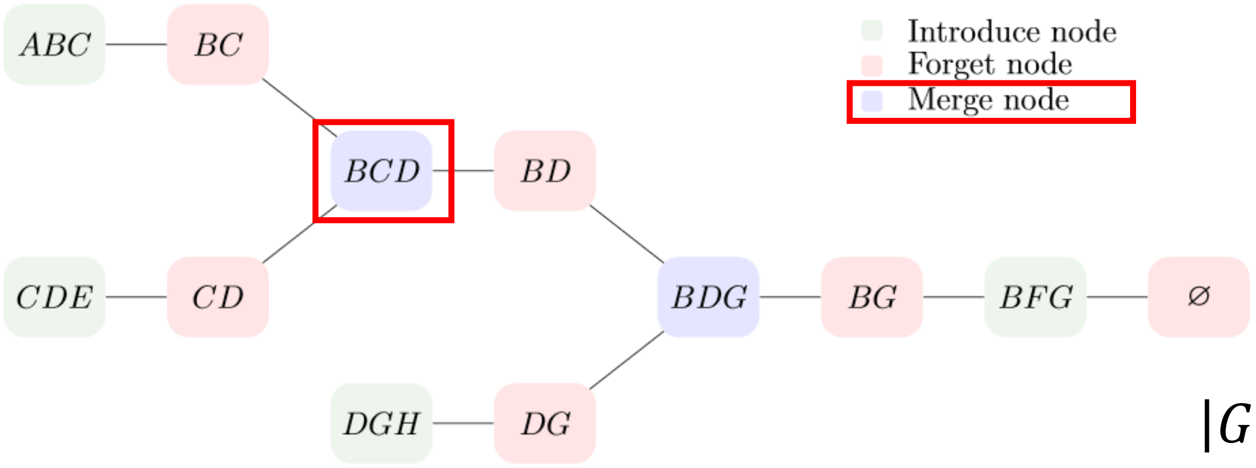


$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+^{\otimes n}\rangle$$

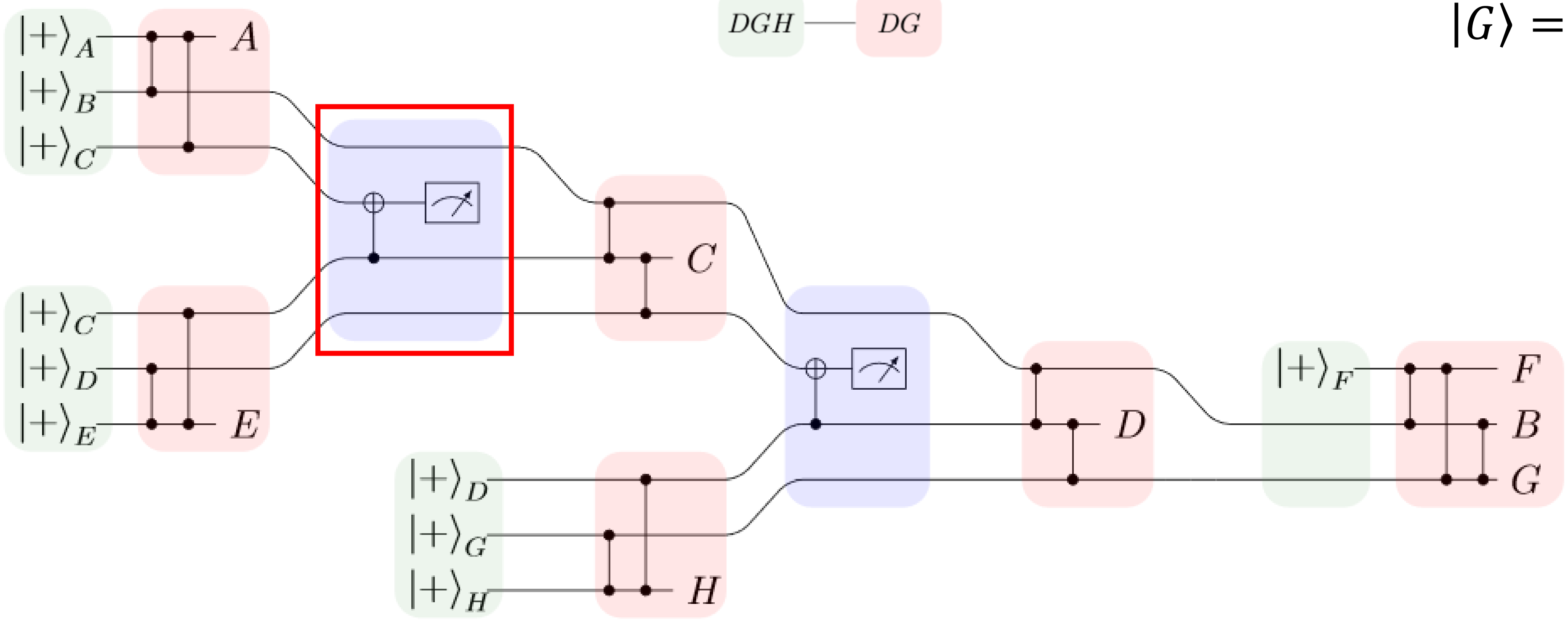


Tree decomposition ↦ circuit

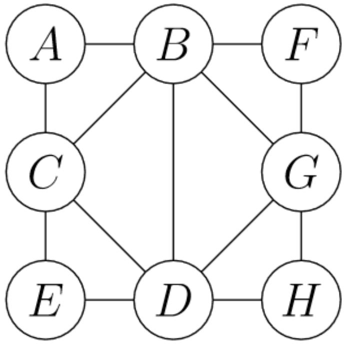


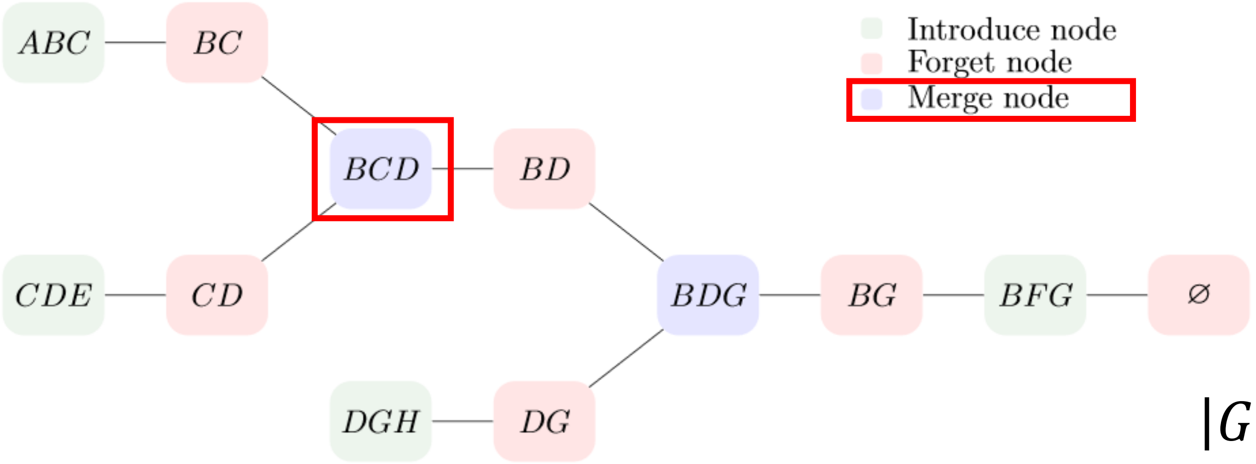


$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+^{\otimes n}\rangle$$

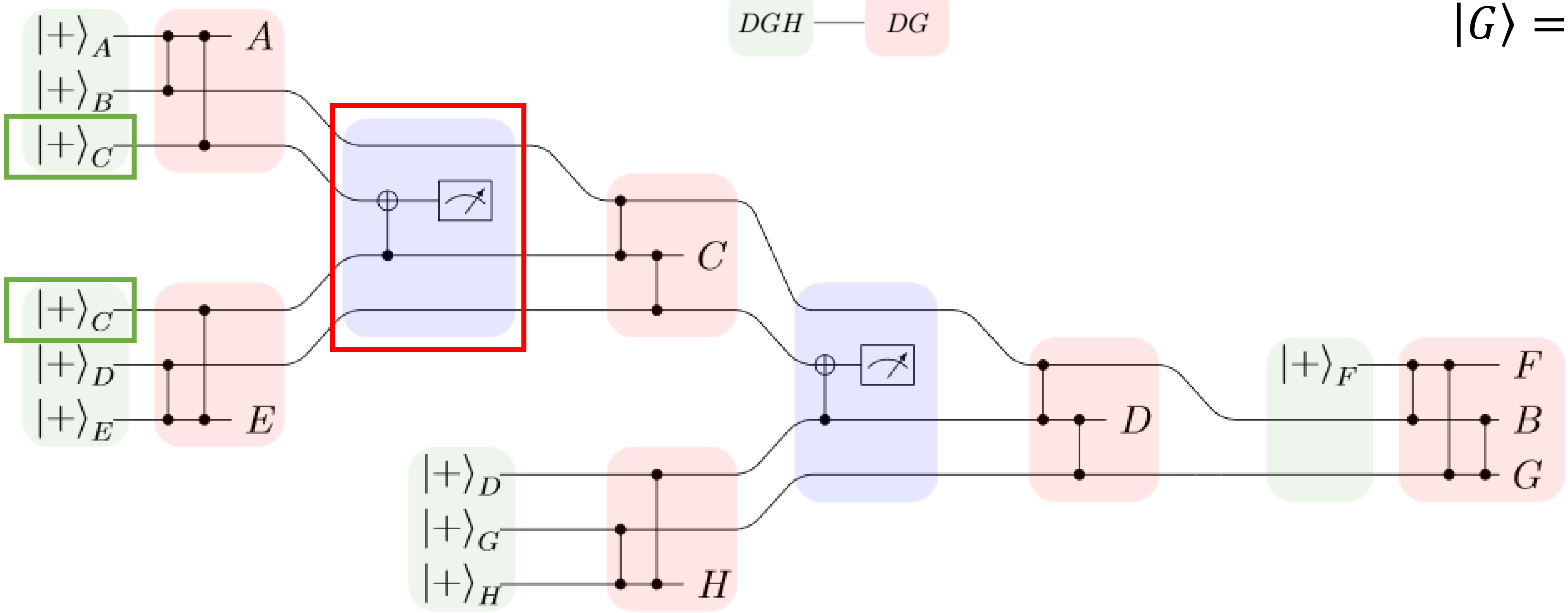


Tree decomposition ↦ circuit

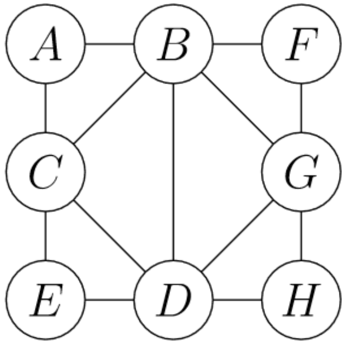


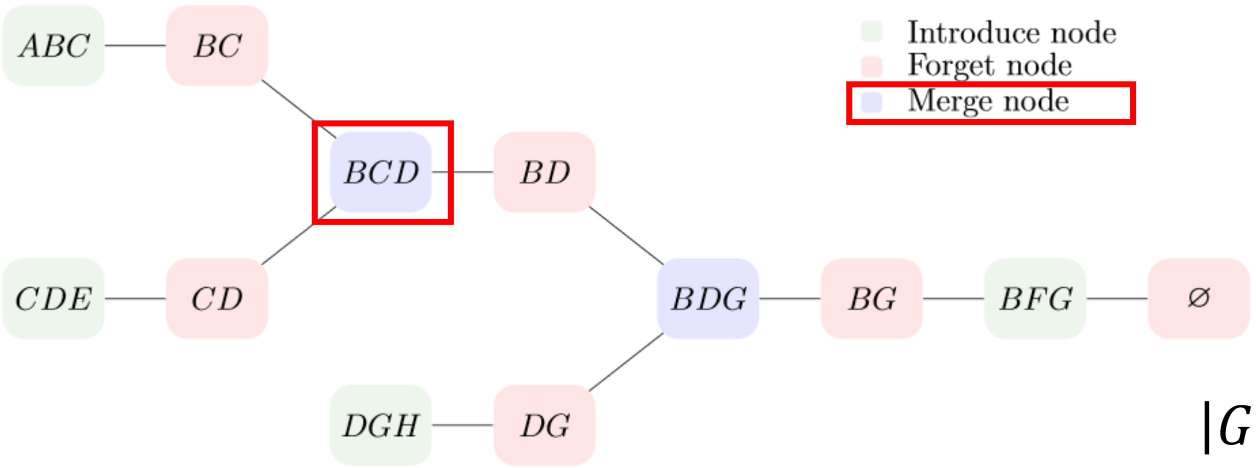


$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab}|+^{\otimes n}\rangle$$

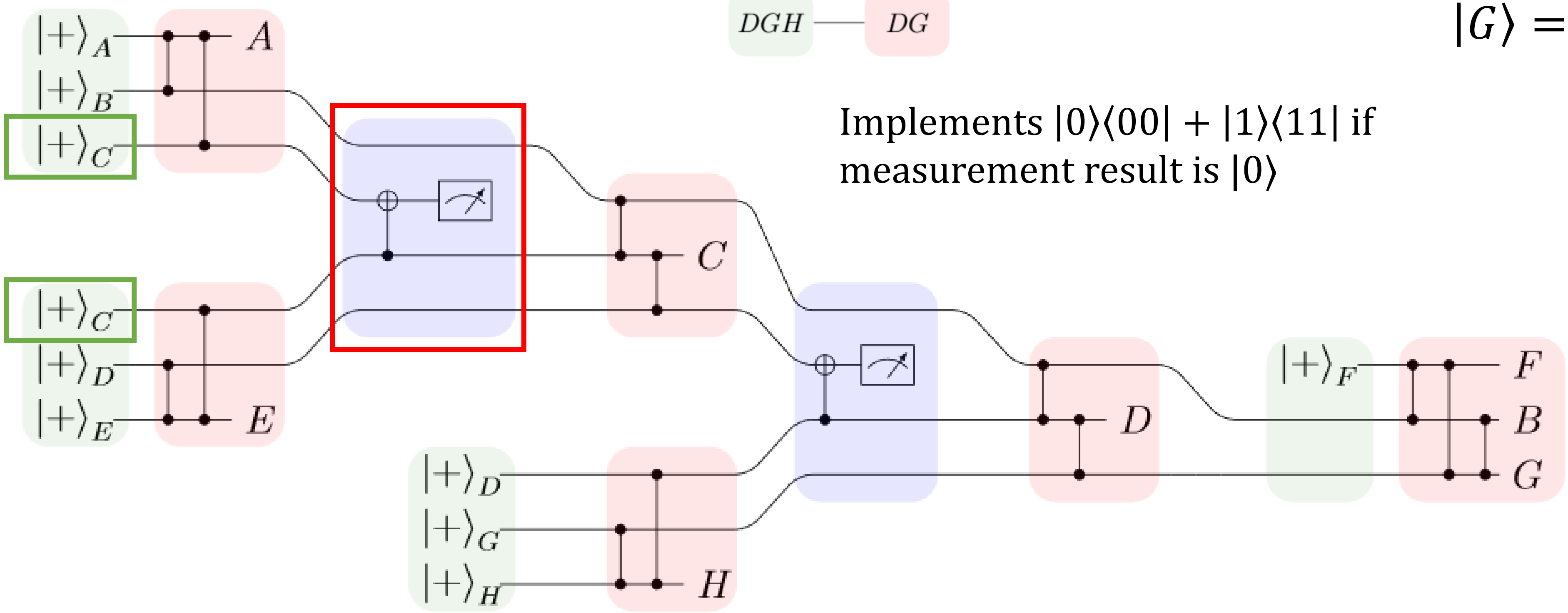


Tree decomposition ↦ circuit



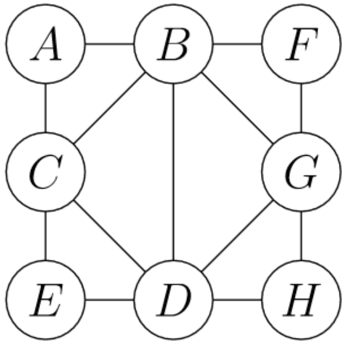


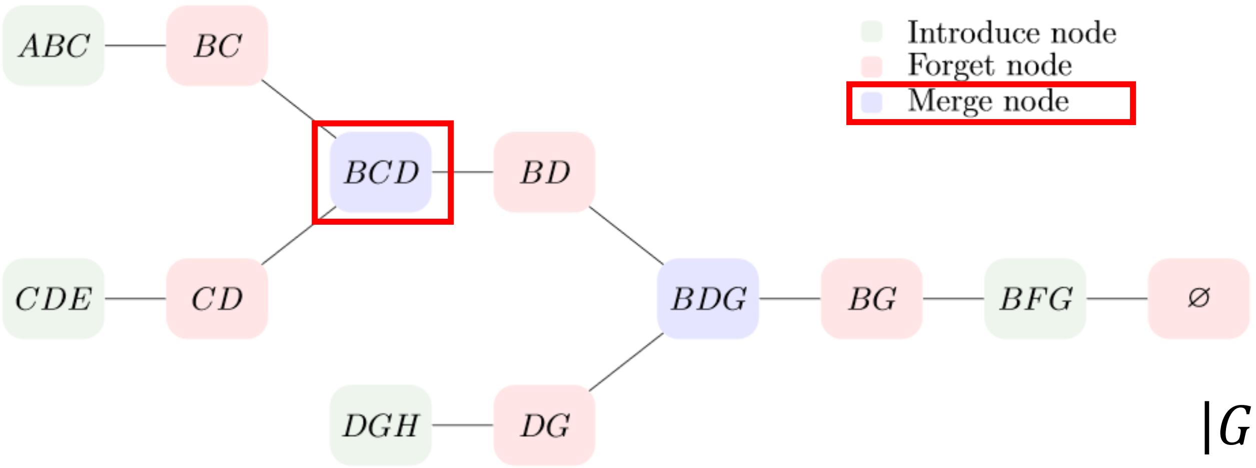
$$|G\rangle = \prod_{ab \in E(G)} CZ_{ab} |+^{\otimes n}\rangle$$



Implements $|0\rangle\langle 00| + |1\rangle\langle 11|$ if measurement result is $|0\rangle$

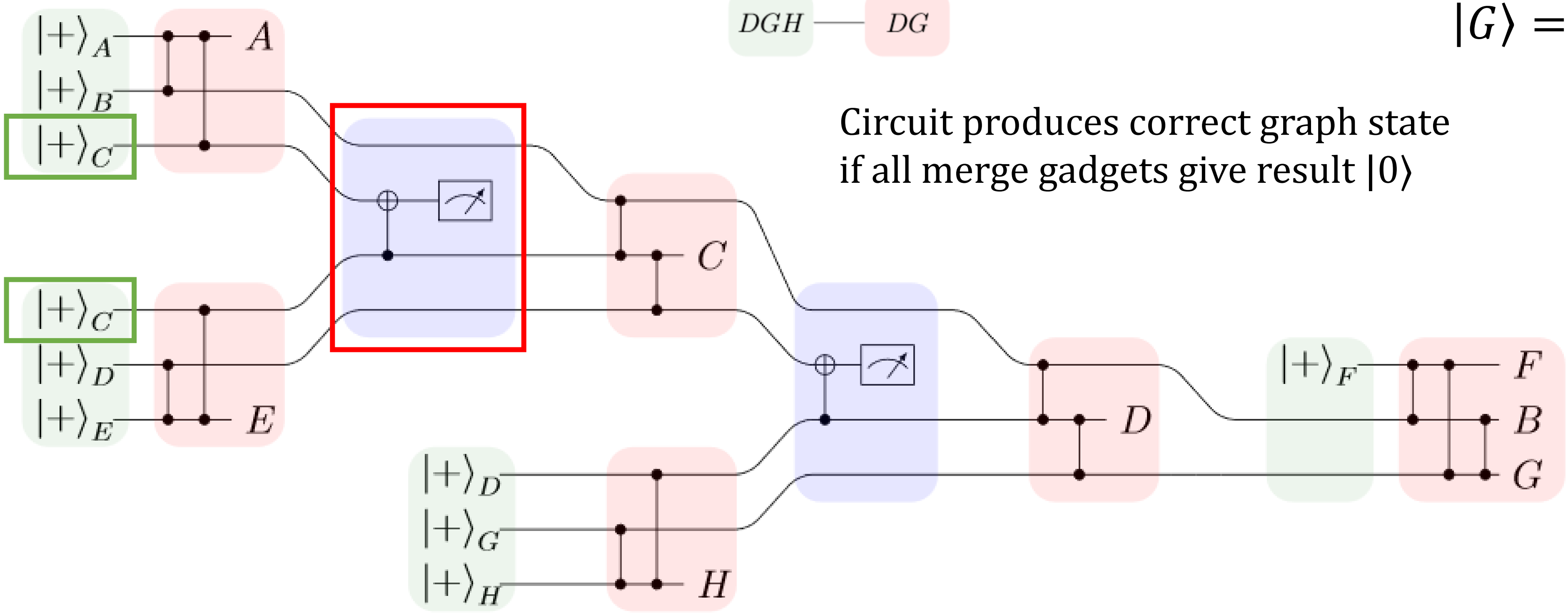
Tree decomposition ↦ circuit



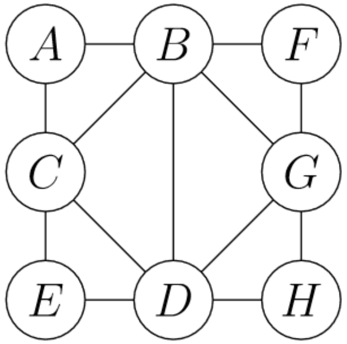


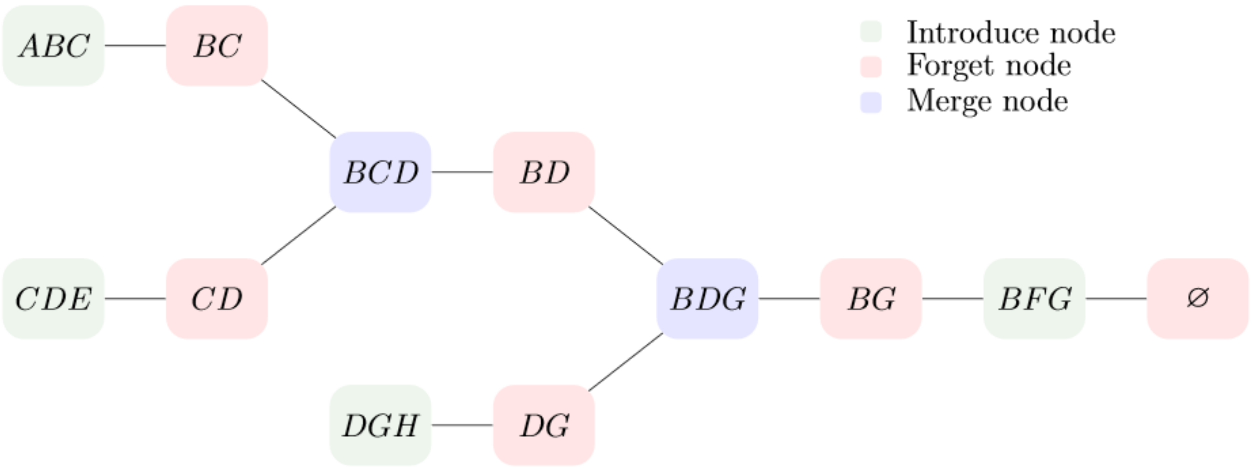
$$|G\rangle = \prod_{ab\in E(G)} CZ_{ab}|+^{\otimes n}\rangle$$

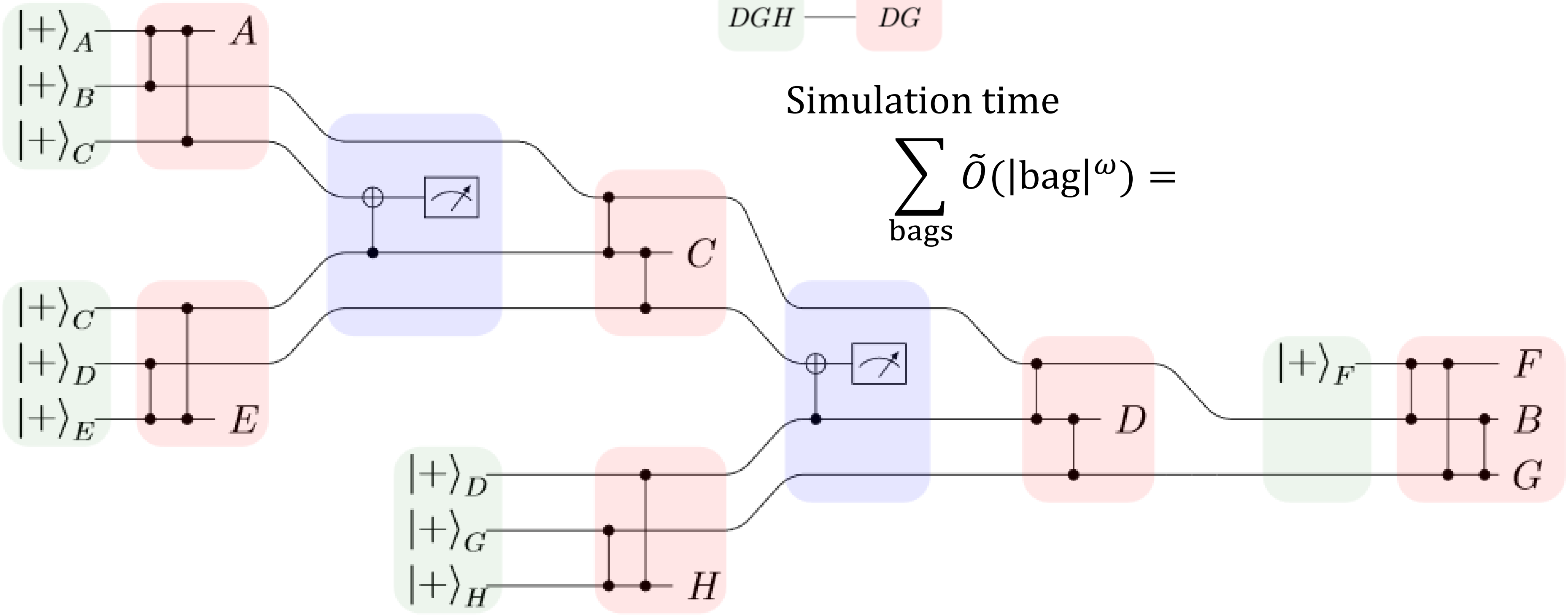
Circuit produces correct graph state if all merge gadgets give result $|0\rangle$



Tree decomposition ↦ circuit



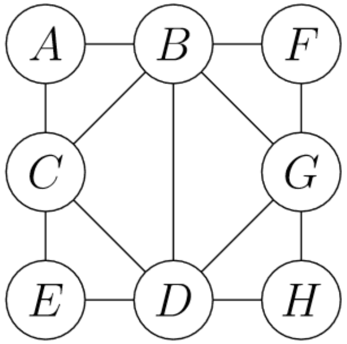


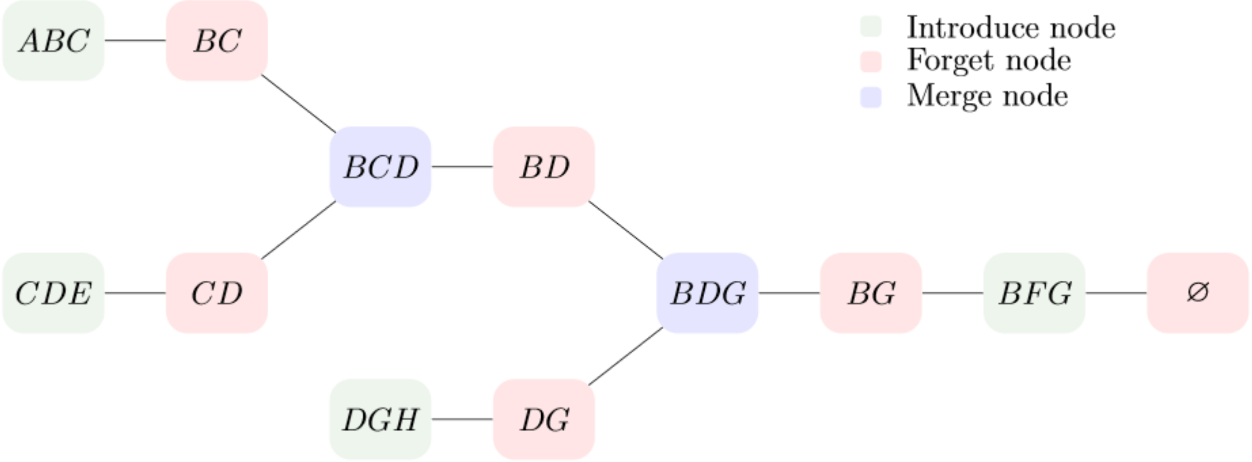


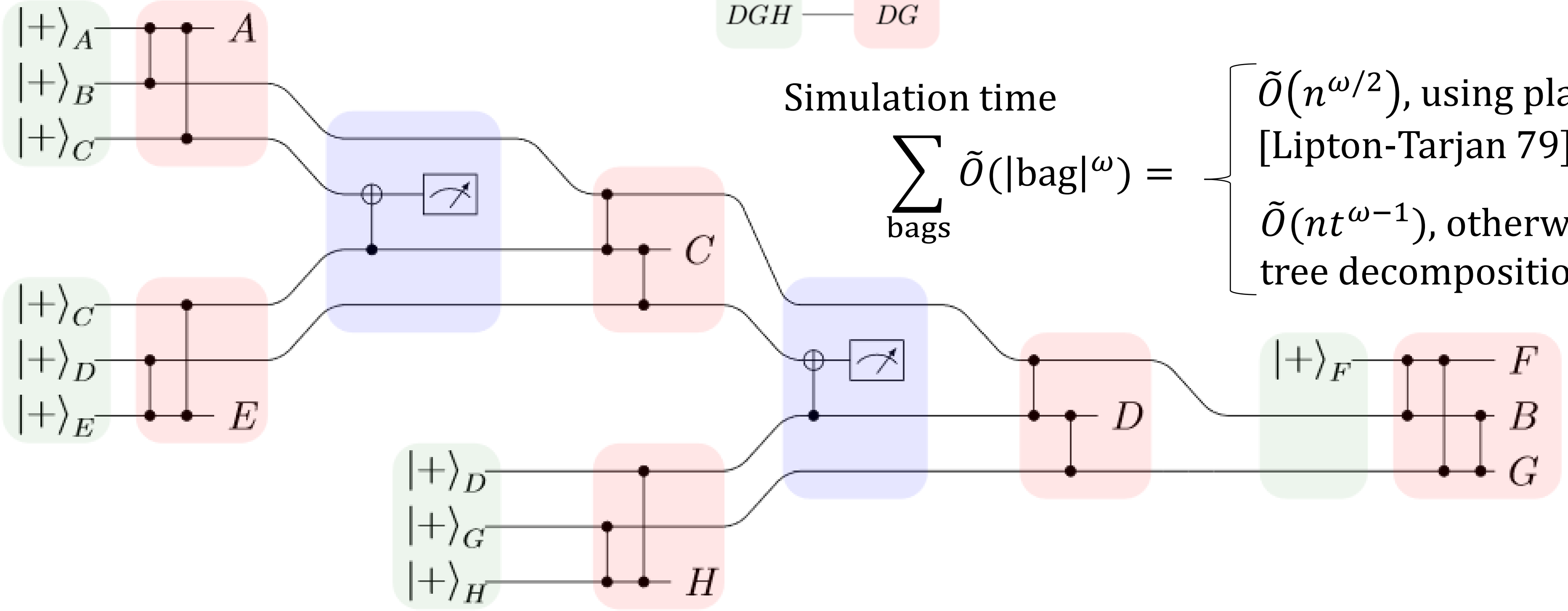
Simulation time

$$\sum_{\text{bags}} \tilde{O}(|\text{bag}|^{\omega}) =$$

Tree decomposition ↦ circuit



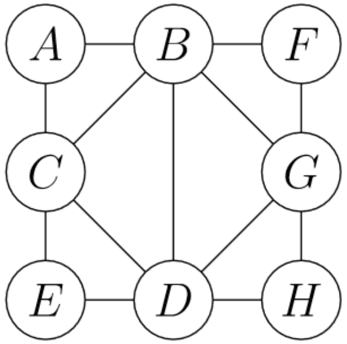


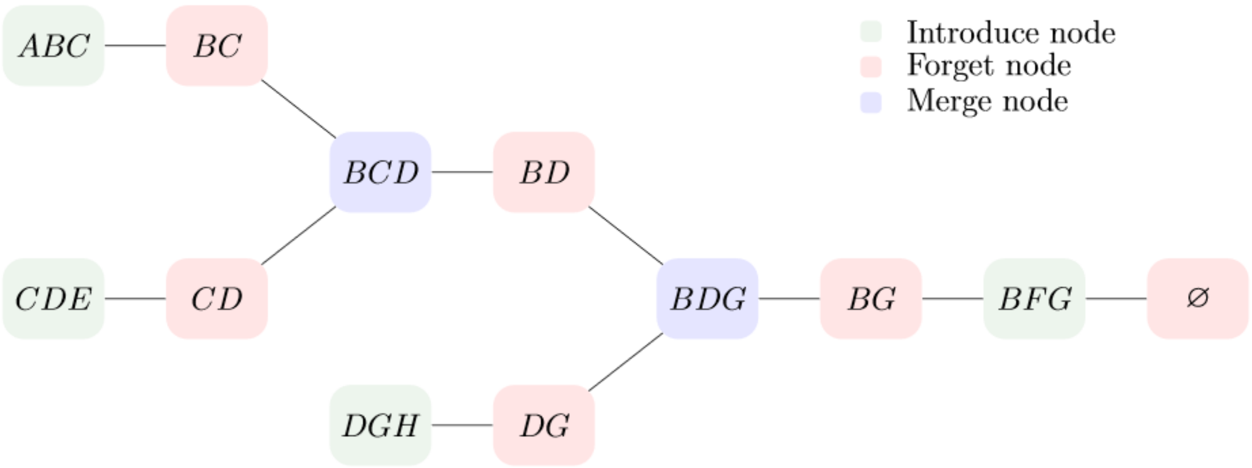


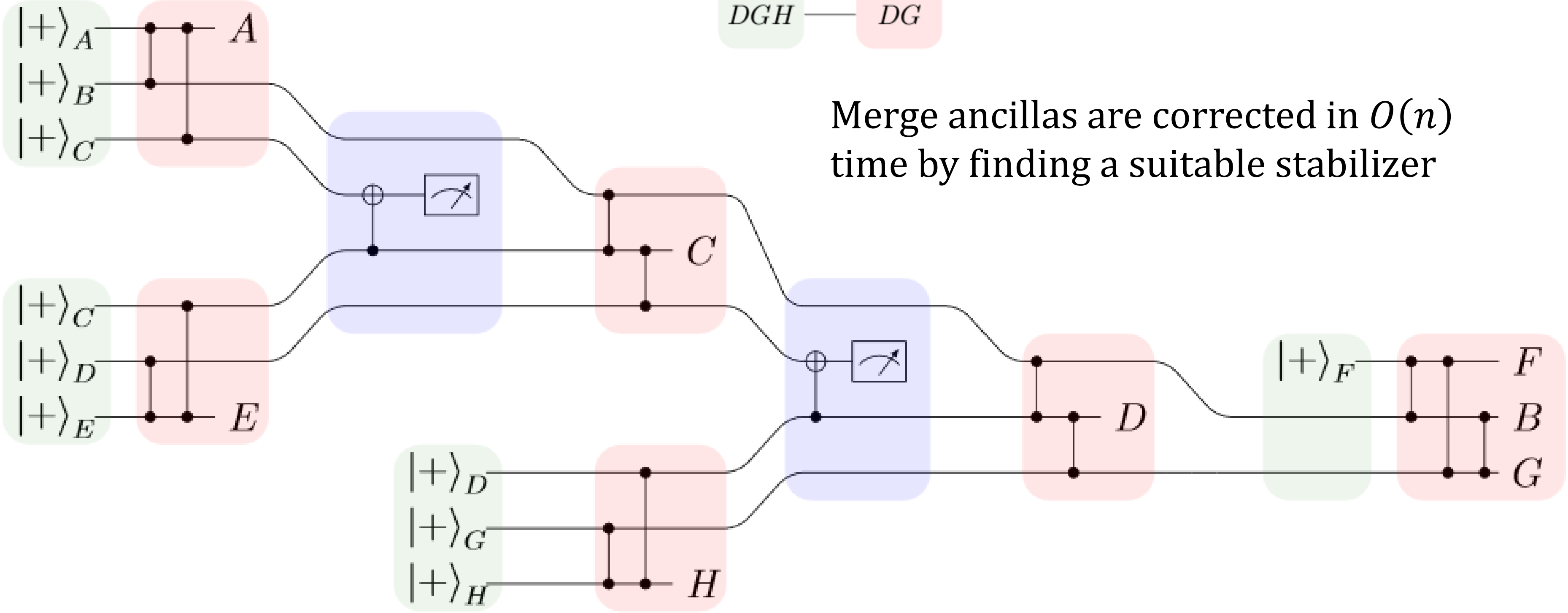
Simulation time

$$\sum_{\text{bags}} \tilde{O}(|\text{bag}|^{\omega}) = \begin{cases} \tilde{O}(n^{\omega/2}), \text{ using planar separators [Lipton-Tarjan 79]} \\ \tilde{O}(nt^{\omega-1}), \text{ otherwise, when given tree decomposition} \end{cases}$$

Tree decomposition ↦ circuit







Merge ancillas are corrected in $O(n)$ time by finding a suitable stabilizer

Other results

Graph state simulation with postselection

Same runtime: $\tilde{O}\left(n^{\omega/2}\right)$

Correction subroutine is equivalent to solving $Ax = b$ over $\mathbb{F}_2$ in time $\tilde{O}\left(n^{\omega/2}\right)$ where A is the adjacency matrix of a planar graph. Extends results of [Alon-Yuster 10] to allow for singular A.

Other results

Graph state simulation with postselection

Same runtime: $\tilde{O}\left(n^{\omega/2}\right)$

Correction subroutine is equivalent to solving $Ax = b$ over $\mathbb{F}_2$ in time $\tilde{O}\left(n^{\omega/2}\right)$ where A is the adjacency matrix of a planar graph. Extends results of [Alon-Yuster 10] to allow for singular A.

Clifford tensor networks

Algorithm for sampling a nonzero element

Summary

Graph state simulation problem

Given G and $P_v \in \{X, Y, Z\}$ for each v, simulate a measurement on $|G\rangle$ in the P_v bases

Theorem

The graph state simulation problem can be solved classically in time $\tilde{O}\left(n^{\omega/2}\right)$ for planar graphs

Theorem

If CZ gates of a Clifford circuit act only on edges of a planar graph, we can sample from the output distribution in time $\tilde{O}(n^{\omega/2} d^{\omega})$

Open problem

Graph state simulation on sparse nonplanar graphs remains a candidate for quantum speedup