Quantum speedups for graph sparsification, graph cut problems and Laplacian solving

Simon Apers[1] Troy Lee[2] Ronald de Wolf[3]

[1]CWI, ULB
[2]University of Technology Sydney
[3]QuSoft, CWI and University of Amsterdam

QIP, February 2021

(arXiv:1911.07306, arXiv:2011.09823)

Graph model

Graph model

undirected, weighted graph $G = (V, E, w)$

n nodes and m edges

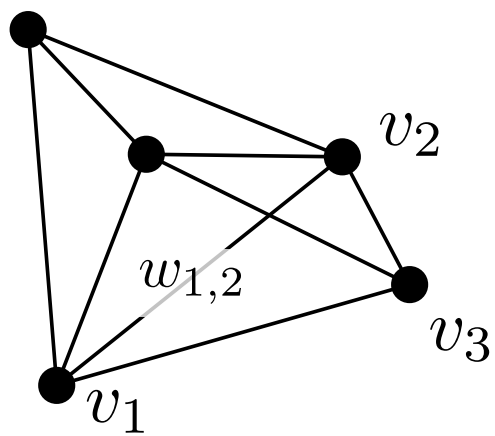
$\in O(n^2)$

Graph model

undirected, weighted graph $G = (V, E, w)$
n nodes and m edges

adjacency list access:

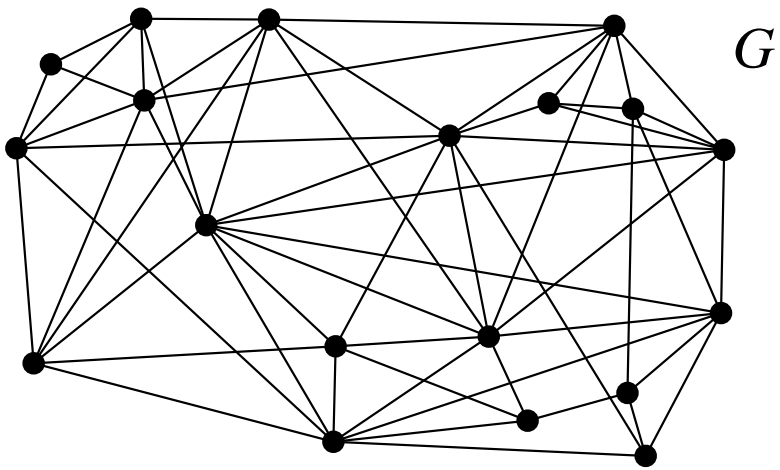
(v_1): $(v_2, w_{1,2})$, $(v_3, w_{1,3})$, ...
(v_2): $(v_3, w_{2,3})$, ...
⋮
(v_n): ...

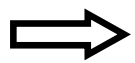


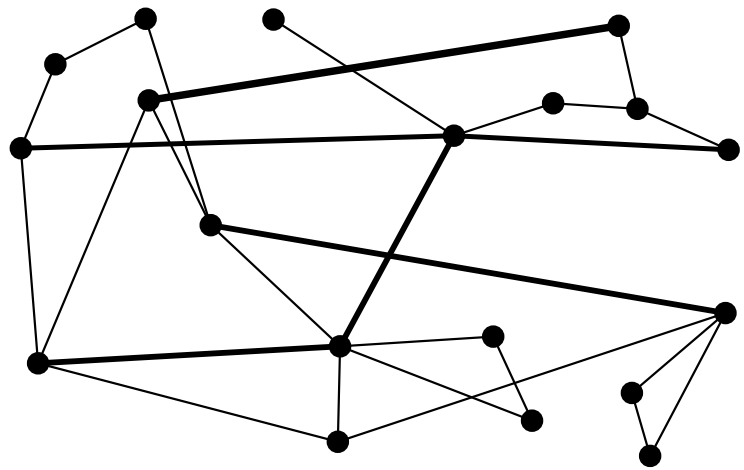
Graph Sparsification

"graph sparsification"

= reduce number of edges, while preserving interesting quantities







$m_G \in O(n^2)$

$m_H << m_G$

Graph Sparsification

"**spectral** sparsification"

Graph Sparsification

"**spectral** sparsification"

= approximately preserve all quadratic forms:

$$x^T L_H x = (1 \pm \varepsilon)\, x^T L_G x, \quad \forall x \in \mathbb{R}^n$$

this includes:

- cut values
- eigenvalues
- random walk properties
- ...

Graph Sparsification

how sparse can we go ?

Graph Sparsification

how sparse can we go ?

Karger '94, Benczúr-Karger '96, Spielman-Teng '04, Batson-Spielman-Srivastava '08:

Theorem

Every graph has ϵ-spectral sparsifier H with a number of edges

$$\widetilde{O}(n/\epsilon^2) \quad << O(n^2)$$

which can be found in time $\widetilde{O}(m)$.

Graph Sparsification

how sparse can we go ?

Karger '94, Benczúr-Karger '96, Spielman-Teng '04, Batson-Spielman-Srivastava '08:

Theorem

Every graph has ϵ-spectral sparsifier H with a number of edges

$$\widetilde{O}(n/\epsilon^2)$$

which can be found in time $\widetilde{O}(m)$.

- building block of many $\widetilde{O}(m)$-time **approximation** algorithms

Graph Sparsification

how sparse can we go ?

Karger '94, Benczúr-Karger '96, Spielman-Teng '04, Batson-Spielman-Srivastava '08:

Theorem

Every graph has ϵ-spectral sparsifier H with a number of edges

$$\widetilde{O}(n/\epsilon^2)$$

which can be found in time $\widetilde{O}(m)$.

- building block of many $\widetilde{O}(m)$-time **approximation** algorithms
- crucial component of Spielman-Teng $\widetilde{O}(m)$-time algorithm for solving **Laplacian system** $Lx = b$

This work:

$m \geq \sqrt{mn}/\varepsilon \geq n/\varepsilon^2$, $\varepsilon \leq \sqrt{\frac{m}{n}}$

This work:

1. **quantum algorithm** to find ϵ-spectral sparsifier H in time

$$\widetilde{O}(\sqrt{mn}/\epsilon)$$

This work:

1. **quantum algorithm** to find ϵ-spectral sparsifier H in time

$$\widetilde{O}(\sqrt{mn}/\epsilon)$$

= vanilla speedup for cut approximation and Laplacian solving

This work:

1. **quantum algorithm** to find ϵ-spectral sparsifier H in time

$$\widetilde{O}(\sqrt{mn}/\epsilon)$$

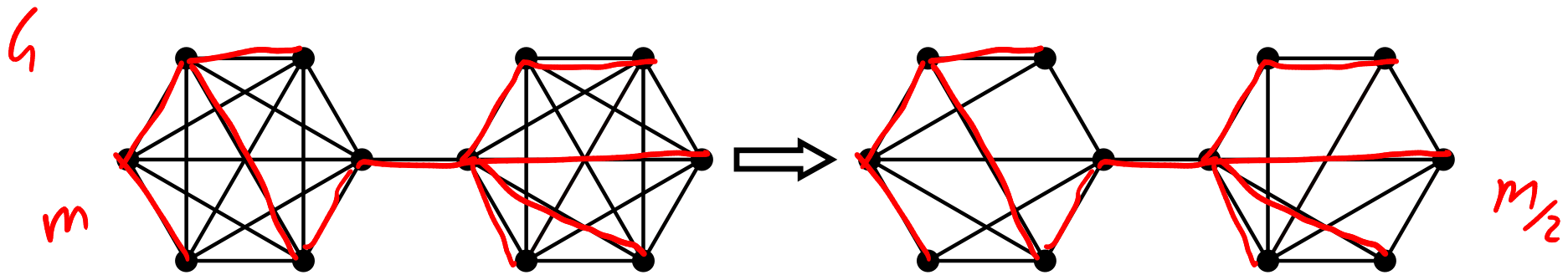
= vanilla speedup for cut approximation and Laplacian solving

2. **quantum algorithm** for finding *exact* minimum cut

Classical Sparsification Algorithm

Iterative sparsification [Koutis-Xu '16]:

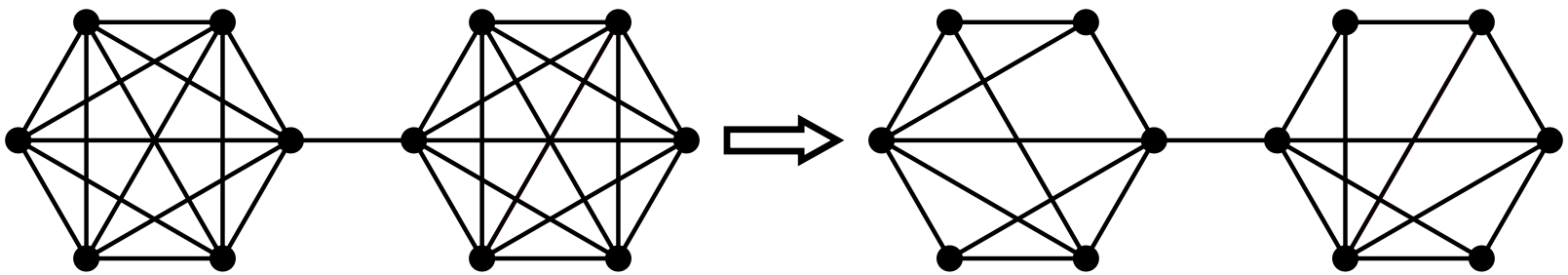
1. construct $\widetilde{O}(1/\epsilon^2)$ "graph spanners" and keep these edges
2. keep any remaining edge with probability $1/2$, and double its weight



Classical Sparsification Algorithm

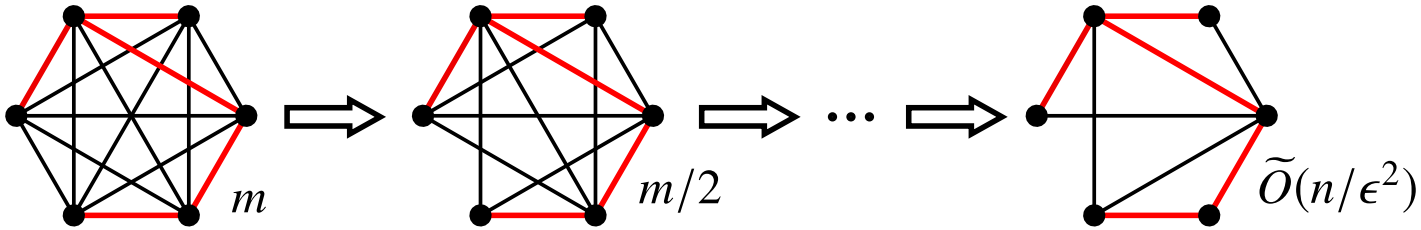
Iterative sparsification [Koutis-Xu '16]:

1. construct $\widetilde{O}(1/\epsilon^2)$ "graph spanners" and keep these edges
2. keep any remaining edge with probability $1/2$, and double its weight

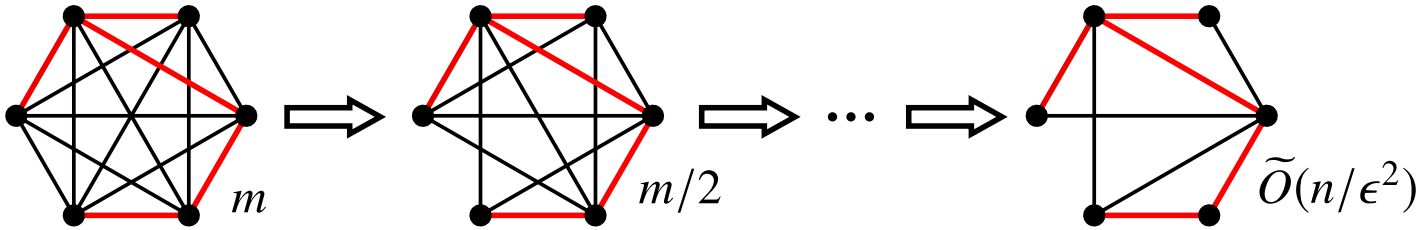


$\rightarrow$ *repeat* $O(\log n)$ *times:* ϵ-spectral sparsifier with $\widetilde{O}(n/\epsilon^2)$ edges

Quantum Sparsification Algorithm



Quantum Sparsification Algorithm



= **quantum spanner algorithm**

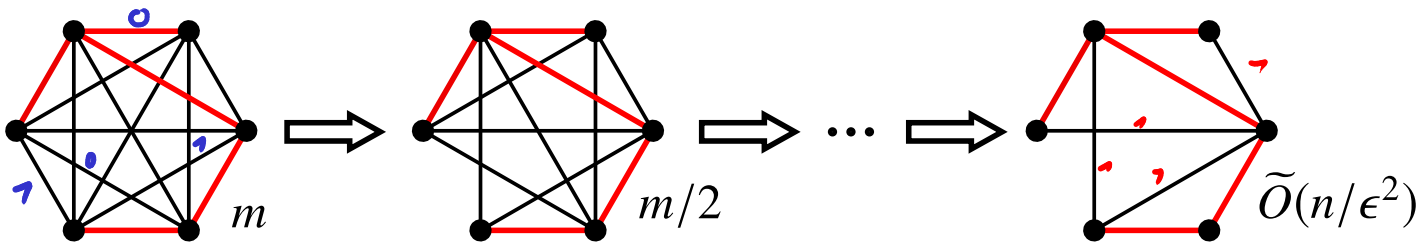
complexity $\widetilde{O}(\sqrt{mn})$

builds on

Thorup-Zwick ('01)

Dürr-Heiligman-Høyer-Mhalla ('06)

Quantum Sparsification Algorithm



= **quantum spanner algorithm**

+ k**-independent oracle**

complexity $\widetilde{O}(\sqrt{mn})$

implicitly downsample the edges

builds on

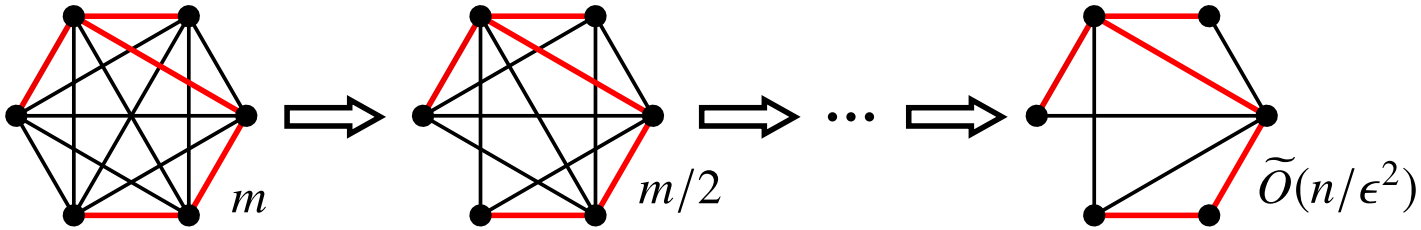
Thorup-Zwick ('01)

Dürr-Heiligman-Høyer-Mhalla ('06)

builds on

Christiani-Pagh-Thorup ('15)

Quantum Sparsification Algorithm



= **quantum spanner algorithm**

complexity $\widetilde{O}(\sqrt{mn})$

builds on

Thorup-Zwick ('01)

Dürr-Heiligman-Høyer-Mhalla ('06)

\+ **k-independent oracle**

implicitly downsample the edges

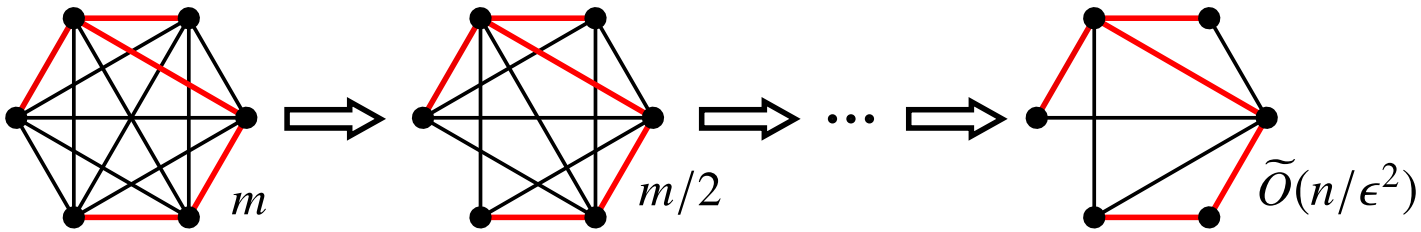
builds on

Christiani-Pagh-Thorup ('15)

↓

$\widetilde{O}(\sqrt{mn}/\epsilon^2)$

Quantum Sparsification Algorithm



= **quantum spanner algorithm**

complexity $\widetilde{O}(\sqrt{mn})$

builds on

Thorup-Zwick ('01)

Dürr-Heiligman-Høyer-Mhalla ('06)

+ **k-independent oracle**

implicitly downsample the edges

builds on

Christiani-Pagh-Thorup ('15)

$\downarrow$

$\widetilde{O}(\sqrt{mn}/\epsilon^2)$

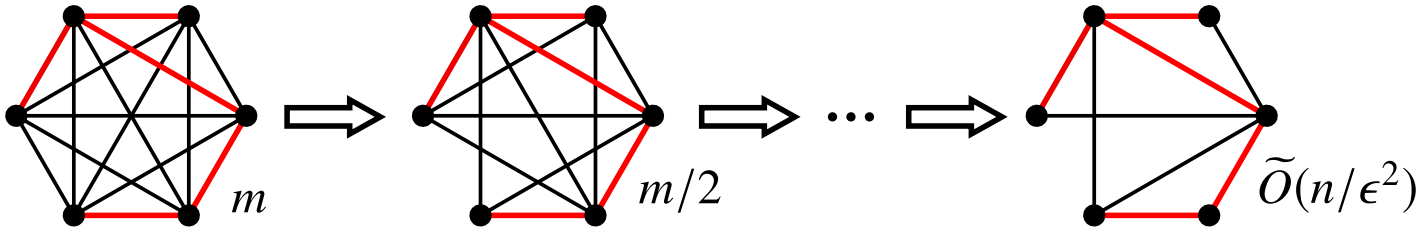
+ **bootstrap trick**

improve ϵ-dependency

builds on

Spielman-Srivastava ('08)

Quantum Sparsification Algorithm



= **quantum spanner algorithm**

complexity $\widetilde{O}(\sqrt{mn})$

builds on

Thorup-Zwick ('01)

Dürr-Heiligman-Høyer-Mhalla ('06)

+ **k-independent oracle**

implicitly downsample the edges

builds on

Christiani-Pagh-Thorup ('15)

$\downarrow$

$\widetilde{O}(\sqrt{mn}/\epsilon^2)$

+ **bootstrap trick**

improve ϵ-dependency

builds on

Spielman-Srivastava ('08)

$\downarrow$

$\widetilde{O}(\sqrt{mn}/\epsilon)$

Quantum Sparsification Algorithm

Theorem

There is a quantum algorithm that constructs an ϵ-spectral sparsifier with $\widetilde{O}(n/\epsilon^2)$ edges in time

$$\widetilde{O}(\sqrt{mn}/\epsilon).$$

Quantum Sparsification Algorithm

Theorem

There is a quantum algorithm that constructs an ϵ-spectral sparsifier with $\widetilde{O}(n/\epsilon^2)$ edges in time

$$\widetilde{O}(\sqrt{mn}/\epsilon).$$

This is tight (up to polylog-factors).

Matching Quantum Lower Bound

intuition:

finding k marked elements among M elements takes

$\Omega(\sqrt{Mk})$ **quantum queries**,

Matching Quantum Lower Bound

intuition:

finding k marked elements among M elements takes

$\Omega(\sqrt{Mk})$ **quantum queries**,

"*hence*"

finding $\widetilde{O}(n/\epsilon^2)$ edges of sparsifier among m edges takes

$\widetilde{\Omega}(\sqrt{mn}/\epsilon)$ **quantum queries**

Matching Quantum Lower Bound

intuition:

finding k marked elements among M elements takes

$\Omega(\sqrt{Mk})$ **quantum queries**,

hence

finding $\widetilde{O}(n/\epsilon^2)$ edges of sparsifier among m edges takes

$\widetilde{\Omega}(\sqrt{mn}/\epsilon)$ **quantum queries**

where “hence”

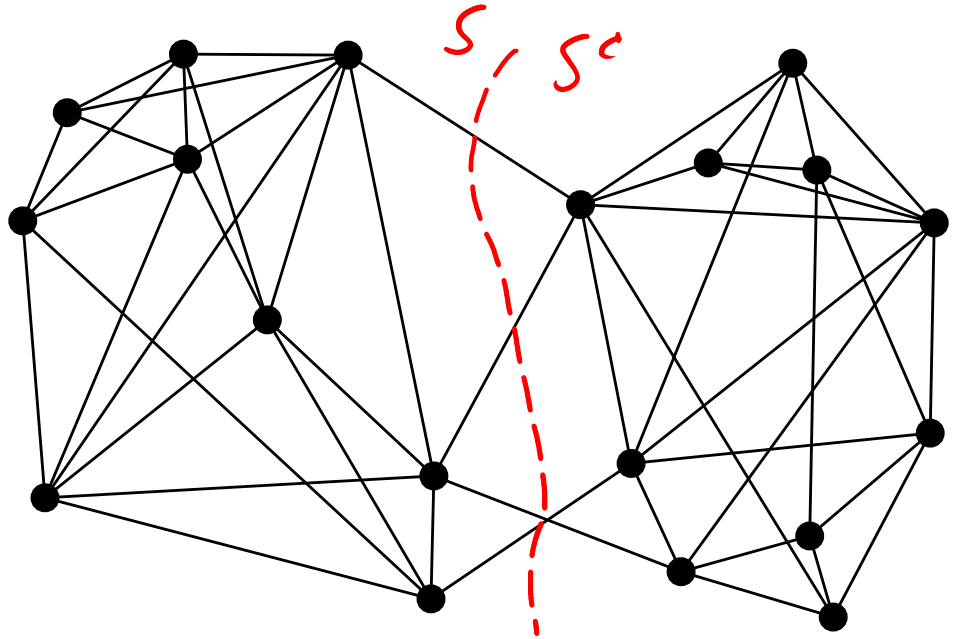
=

hiding an unsparsifiable graph [Andoni-Chen-Krauthgamer-Qin-Woodruff-Zhang ’16]

+ quantum lower bound for relational problem [Belov-Lee ’20]

Applications

e.g., MIN CUT:



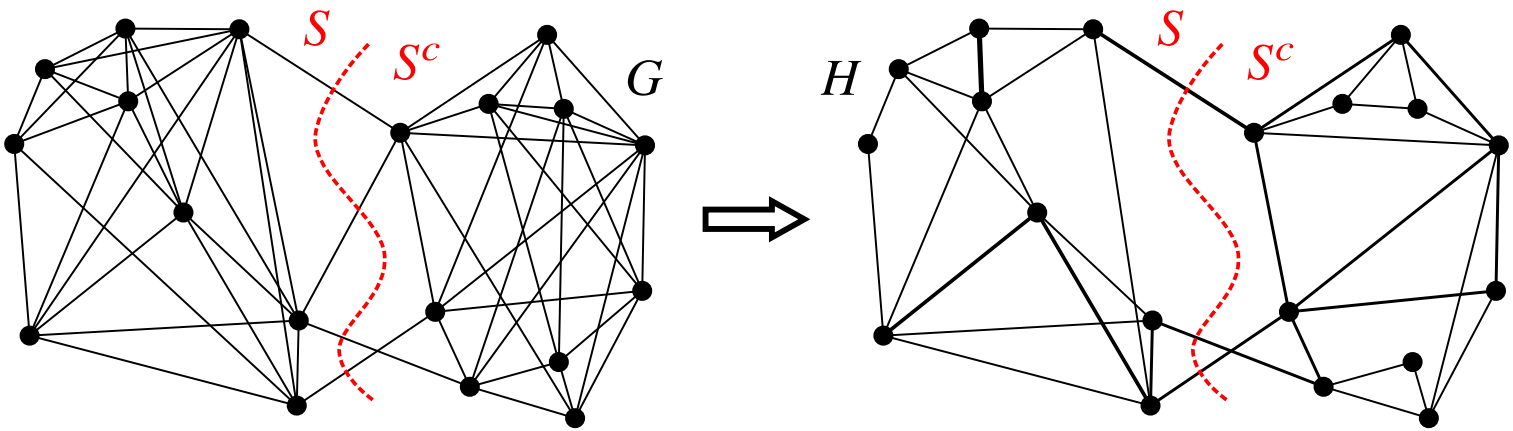
find cut (S, S^c) that minimizes cut value

$$\text{cut}_G(S) = \sum_{i \in S, j \notin S} w_{i,j} \quad = 4$$

Applications

MIN CUT of ϵ-spectral sparsifier H
gives ϵ-approximation of MIN CUT of G

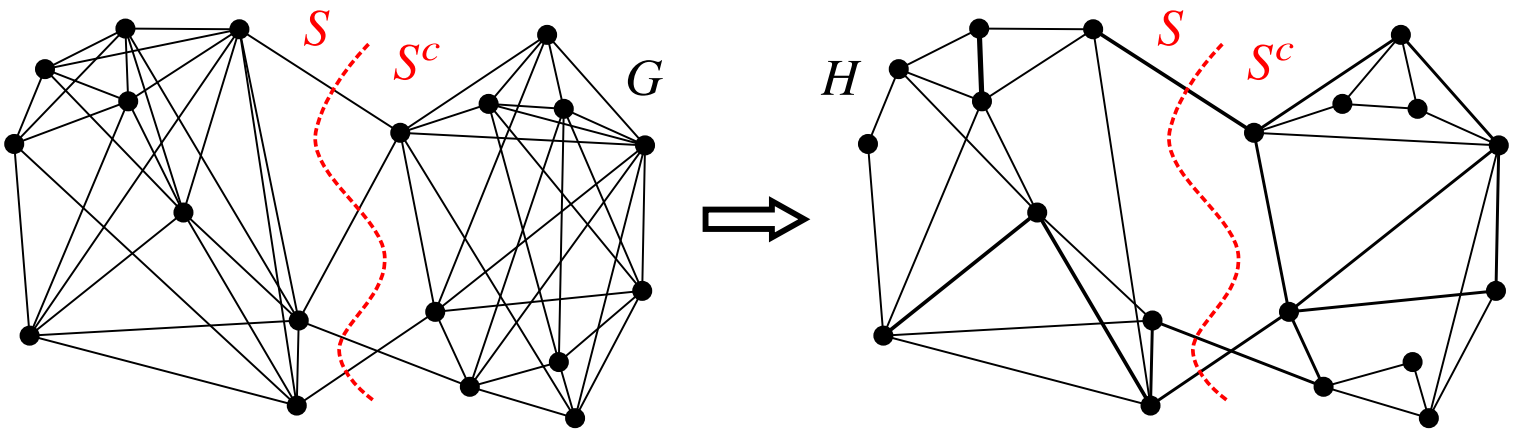
$O(n^2)$



$\tilde{O}(n/\epsilon^2)$

Applications

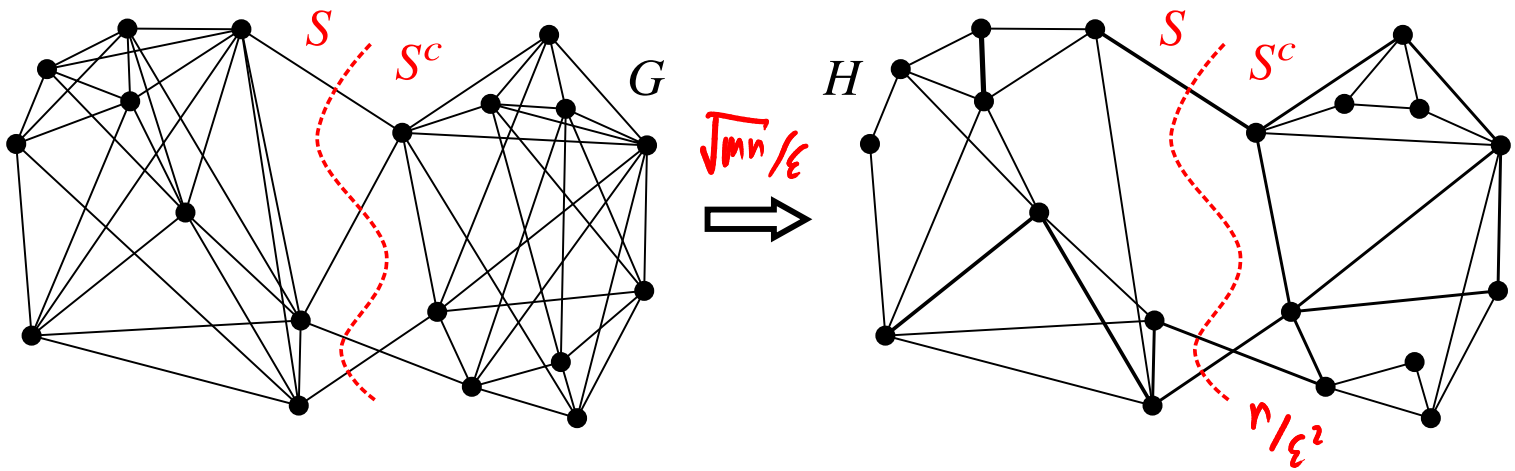
MIN CUT of ϵ-spectral sparsifier H
gives ϵ-approximation of MIN CUT of G



classically:
can find MIN CUT in time $\widetilde{O}(m)$ (Karger '00)

Applications

MIN CUT of ϵ-spectral sparsifier H
gives ϵ-approximation of MIN CUT of G

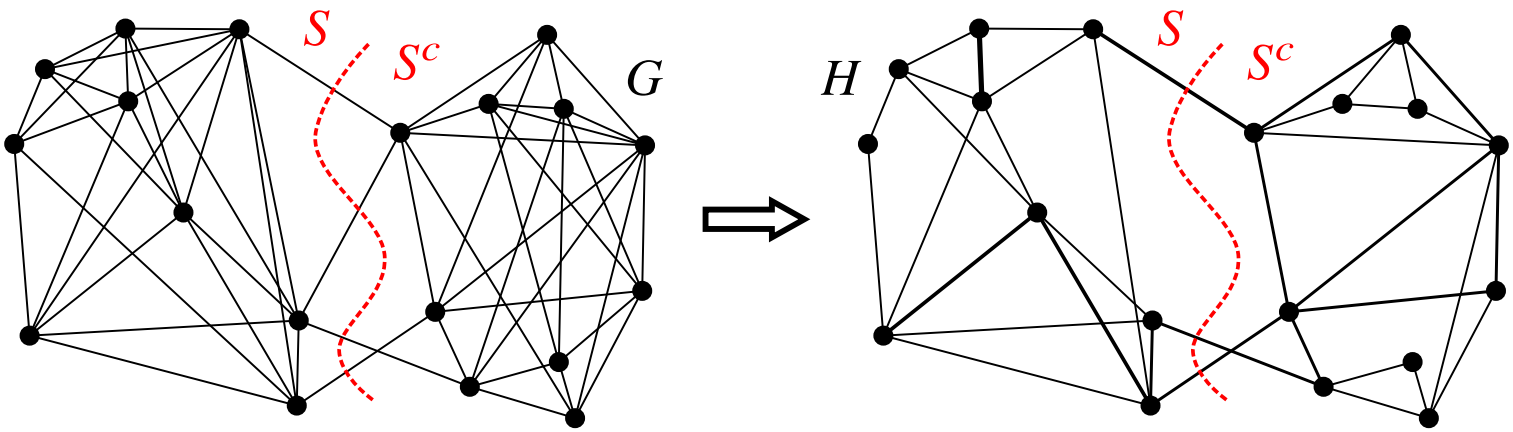


classically:
can find MIN CUT in time $\widetilde{O}(m)$ (Karger '00)

quantum:
quantum algorithm to create sparsifier H in $\widetilde{O}(\sqrt{mn}/\epsilon)$
+ **classical** MIN CUT on H in $\widetilde{O}(n/\epsilon^2)$ (Karger '00)

Applications

MIN CUT of ϵ-spectral sparsifier H
gives ϵ-approximation of MIN CUT of G



classically:
can find MIN CUT in time $\widetilde{O}(m)$ (Karger '00)

quantum:
quantum algorithm to create sparsifier H in $\widetilde{O}(\sqrt{mn}/\epsilon)$
+ **classical** MIN CUT on H in $\widetilde{O}(n/\epsilon^2)$ (Karger '00)

= $\widetilde{O}(\sqrt{mn}/\epsilon)$ quantum algorithm

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

- **cut problems:**
  - min cut, max cut $(\epsilon > .878)$, sparsest cut $(\epsilon \in \Omega(\sqrt{\log n}))$, . . .

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

- **cut problems:**
  - min cut, max cut $(\epsilon > .878)$, sparsest cut $(\epsilon \in \Omega(\sqrt{\log n}))$, ...
- **Laplacian systems** $L_G x = b$**:**
  - given sparse access to Laplacian or SDD matrix L, explicitly outputs ϵ-approximation to x

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

- **cut problems:**
  - min cut, max cut $(\epsilon > .878)$, sparsest cut $(\epsilon \in \Omega(\sqrt{\log n}))$, ...
- **Laplacian systems** $L_G x = b$**:**
  - given sparse access to Laplacian or SDD matrix L, explicitly outputs ϵ-approximation to x
  - solution to $L_H x = b$ approximates solution to $L_G x = b$

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

- **cut problems:**
  - min cut, max cut $(\epsilon > .878)$, sparsest cut $(\epsilon \in \Omega(\sqrt{\log n}))$, ...
- **Laplacian systems** $L_G x = b$**:**
  - given sparse access to Laplacian or SDD matrix L, explicitly outputs ϵ-approximation to x
  - solution to $L_H x = b$ approximates solution to $L_G x = b$
  - applications in combinatorial optimization, machine learning, ...

Applications

$\rightarrow$ general blueprint:

1. Use **quantum** algorithm to create sparsifier H in time $\widetilde{O}(\sqrt{mn}/\epsilon)$
2. Run **classical** algorithm on H in time $\widetilde{O}(n/\epsilon^2)$

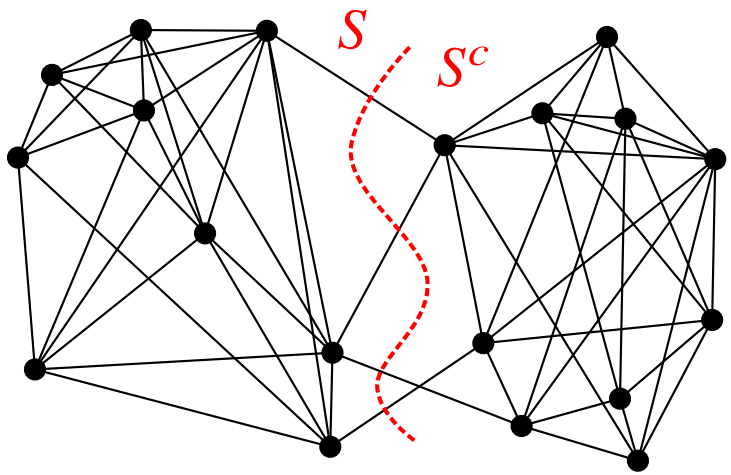
= $\widetilde{O}(\sqrt{mn}/\epsilon)$**-time** quantum algorithms for ϵ**-approximating**

- **cut problems:**
  - min cut, max cut $(\epsilon > .878)$, sparsest cut $(\epsilon \in \Omega(\sqrt{\log n}))$, ...
- **Laplacian systems** $L_G x = b$**:**
  - given sparse access to Laplacian or SDD matrix L, explicitly outputs ϵ-approximation to x
  - solution to $L_H x = b$ approximates solution to $L_G x = b$
  - applications in combinatorial optimization, machine learning, ...

Can we get exact quantum algorithms?

Quantum algorithm for MIN CUT

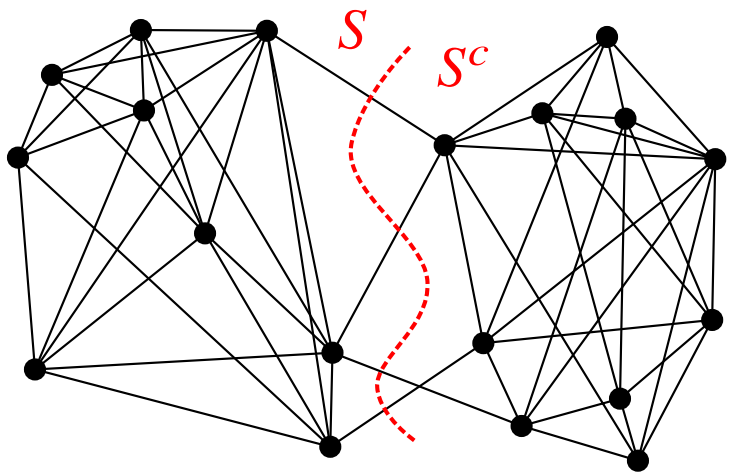
back to MIN CUT:



classical: *exact* MIN CUT in time $\widetilde{O}(m)$ (Karger '00)

Quantum algorithm for MIN CUT

back to MIN CUT:



classical: *exact* MIN CUT in time $\widetilde{O}(m)$ (Karger '00)

quantum: ϵ-*approximate* MIN CUT in time $\widetilde{O}(\sqrt{mn}/\epsilon)$:(

Quantum algorithm for MIN CUT

input size = m input size = n^2

Results in adjacency *list* model and adjacency *matrix* model:

Theorem

Quantum algorithm for MIN CUT

Results in adjacency *list* model and adjacency *matrix* model:

Theorem

- *There exists a family of* ***weighted graphs*** *with* $\Omega(n^2)$ *edges for which the quantum query complexity of exact* MIN CUT *is* $\Omega(n^2)$ *in both models.* → no quantum speedup!

Quantum algorithm for MIN CUT

Results in adjacency *list* model and adjacency *matrix* model:

Theorem

- *There exists a family of* ***weighted graphs*** *with* $\Omega(n^2)$ *edges for which the quantum query complexity of exact* MIN CUT *is* $\Omega(n^2)$ *in both models.*
- *For* ***unweighted graphs****, exact* MIN CUT *has quantum query complexity*
  - $\widetilde{O}(\sqrt{mn})$ *in adjacency list model*
  - $\widetilde{O}(n^{3/2})$ *in adjacency matrix model*

Quantum algorithm for MIN CUT

Results in adjacency *list* model and adjacency *matrix* model:

Theorem

- *There exists a family of* ***weighted graphs*** *with* $\Omega(n^2)$ *edges for which the quantum query complexity of exact* MIN CUT *is* $\Omega(n^2)$ *in both models.*
- *For* ***unweighted graphs****, exact* MIN CUT *has quantum query complexity*
  - $\widetilde{O}(\sqrt{mn})$ *in adjacency list model* ← improves $\widetilde{O}(\sqrt{mn}/\varepsilon)$
  - $\widetilde{O}(n^{3/2})$ *in adjacency matrix model*

  and time complexity $\widetilde{O}(n^{3/2})$ *in both models.*

Quantum algorithm for MIN CUT

Results in adjacency *list* model and adjacency *matrix* model:

Theorem

- *There exists a family of* ***weighted graphs*** *with* $\Omega(n^2)$ *edges for which the quantum query complexity of exact* MIN CUT *is* $\Omega(n^2)$ *in both models.*
- *For* ***unweighted graphs****, exact* MIN CUT *has quantum query complexity*
  - $\widetilde{O}(\sqrt{mn})$ *in adjacency list model*
  - $\widetilde{O}(n^{3/2})$ *in adjacency matrix model*

  and time complexity $\widetilde{O}(n^{3/2})$ *in both models.*

tight

gap

Lower bounds for unweighted graphs (from connectivity [DHHM'06]):

$\Omega(n)$ (list) or $\Omega(n^{3/2})$ (matrix)

Quantum algorithm for MIN CUT

Upper bounds on quantum query complexity:

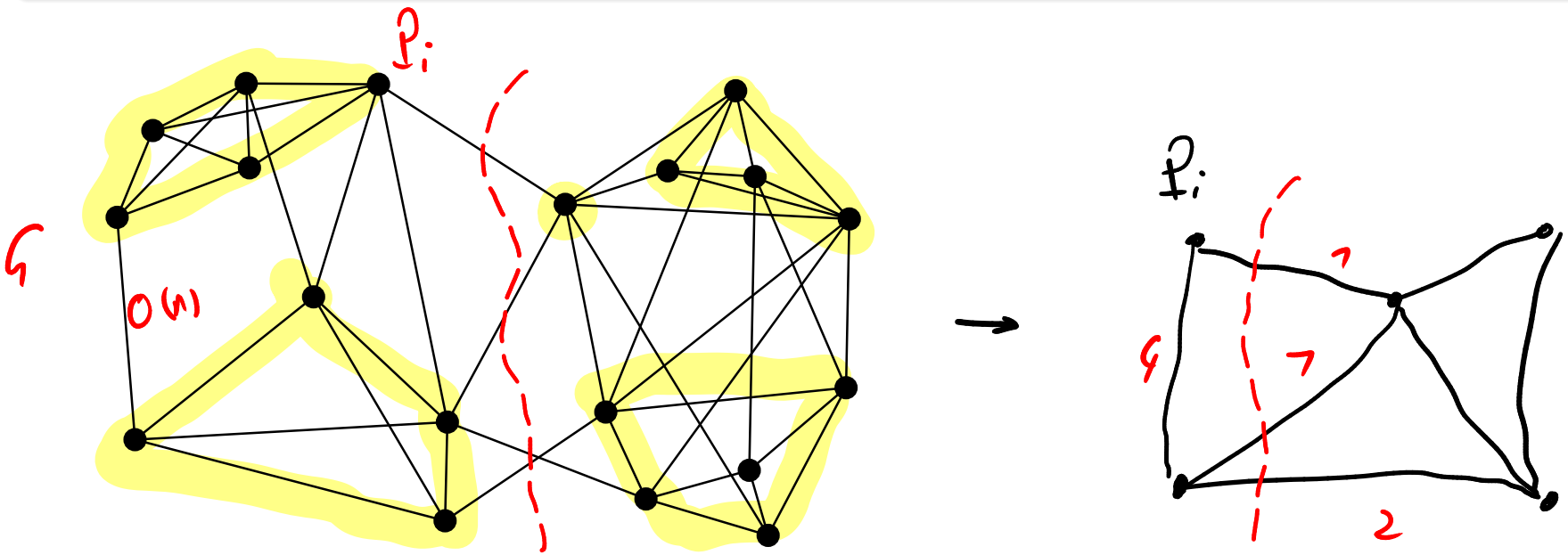
Quantum algorithm for MIN CUT

Upper bounds on quantum query complexity:

Theorem (Kawarabayashi-Thorup 2014, Rubinstein-Schramm-Weinberg 2017)

For unweighted G, there exists partition $V = P_1 \cup \cdots \cup P_k$ such that

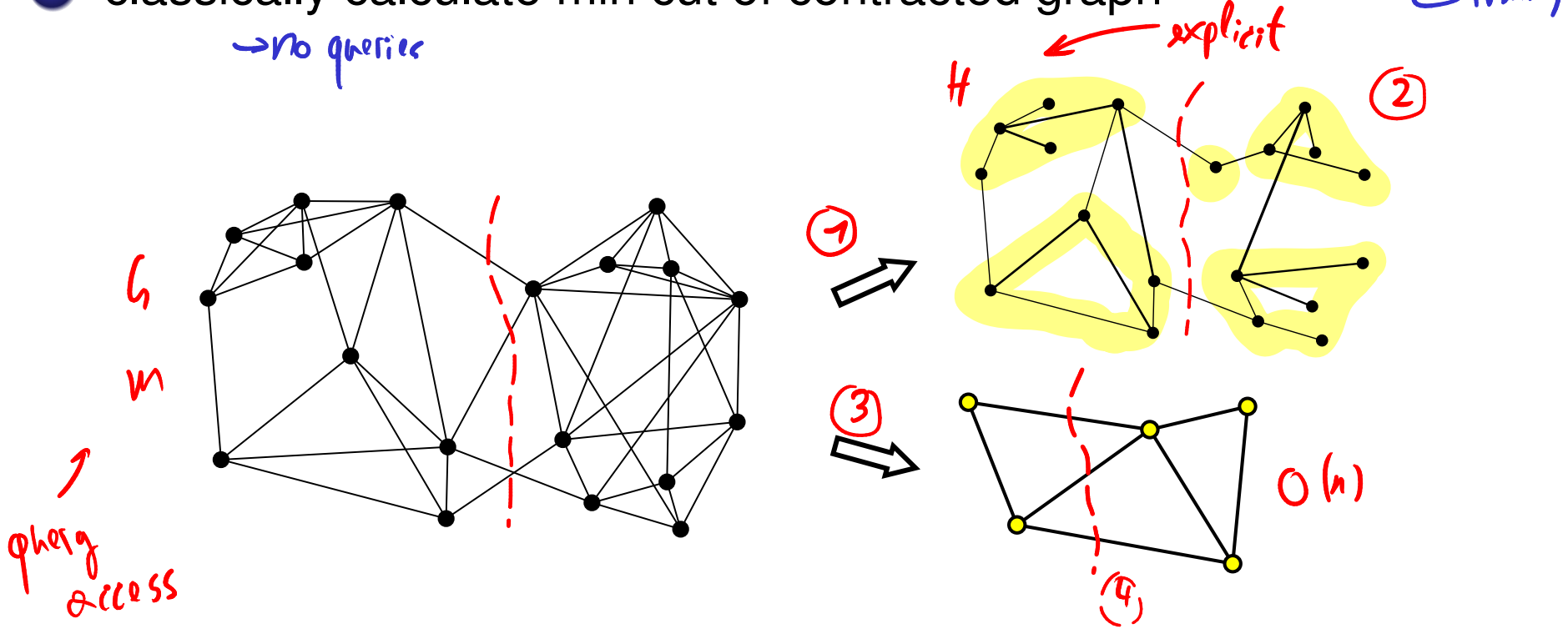
1. *there are $O(n)$ edges between partitions,*
2. *partition "respects" all near-minimum cuts.*



Quantum algorithm for MIN CUT

quantum query algorithm in list model:

1. construct $\epsilon = 1/10$-sparsifier H → $\tilde{O}(\sqrt{mn})$
   → min cut of G is near-minimum cut of H
2. learn partition $V = P_1 \cup \cdots \cup P_k$ of H → no queries
   → respects min cut of G
3. Grover search over m edges to find $O(n)$ edges *of* G between P_i's
4. classically calculate min cut of contracted graph → $O(\sqrt{mn})$



Open questions:

Open questions:

- matching **lower bounds for approximation algorithms?**
  - e.g., $\Omega(\sqrt{mn}/\epsilon)$ for approximate min cut or Laplacian solving?

Open questions:

- matching **lower bounds for approximation algorithms?**
  - e.g., $\Omega(\sqrt{mn}/\epsilon)$ for approximate min cut or Laplacian solving?
- **unweighted graphs:**
  - $\widetilde{O}(n/\epsilon^2)$ for sparsification?
  - $\widetilde{O}(n)$ for unweighted min cut in adjacency list model?

$\Omega(n) \rightarrow \tilde{O}(\sqrt{mn})$

Open questions:

- matching **lower bounds for approximation algorithms?**
  - e.g., $\Omega(\sqrt{mn}/\epsilon)$ for approximate min cut or Laplacian solving?
- **unweighted graphs:**
  - $\widetilde{O}(n/\epsilon^2)$ for sparsification?
  - $\widetilde{O}(n)$ for unweighted min cut in adjacency list model?

thank you! stay safe!

$\widetilde{O}(n^{3/2}/\epsilon) < \widetilde{O}(n^2)$