Learning graphs and quantum query algorithms

Alexander Belov
University of Latvia

IEGULDĪJUMS TAVĀ NĀKOTNĒ

This work has been supported by the European Social Fund within the project "Support for Doctoral Studies at University of Latvia"

Outline

Query Complexity
Adversary Bound
Learning Graphs
Element Distinctness

- Adversary Upper Bound
- Adversary Lower Bound

Various Problems

- Triangle Detection
- k-distinctness

Summary and Future Work

Query complexity

Query complexity

Query complexity

Query complexity

17th bit of the input?

Query complexity

17th bit of the input?

It's 0

Query complexity

17th bit of the input?

It's 0

What about 41st?

Query complexity

17th bit of the input?

It's 0

What about 41st?

It's 1

Query complexity

17th bit of the input?

It's 0

What about 41st?

It's 1

$\boxed{\text{Query Complexity} = \text{Number of queries in the worst case}}$

Query complexity

17th bit of the input?

It's 0

What about 41st?

It's 1

$\boxed{\text{Query Complexity} = \text{Number of queries in the worst case}}$

- Useful if input queries are very expensive

Query complexity

17th bit of the input?

It's 0

What about 41st?

It's 1

Query Complexity = Number of queries in the worst case

- Useful if input queries are very expensive
- We can prove lower bounds for this model

Query complexity

17th bit of the input? ←

→ It's 0

What about 41st? ←

→ It's 1

$\boxed{\text{Query Complexity} = \text{Number of queries in the worst case}}$

- Useful if input queries are very expensive
- We can prove lower bounds for this model
- May lead to new insights

Lower and upper bounds

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

Lower and upper bounds

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

- Polynomial bound (Beals *et al.*, 1998)

Lower and upper bounds

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

- Adversary bound (Ambainis, 2000)
- Polynomial bound (Beals *et al.*, 1998)

Lower and upper bounds

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

- General (negative-weight) adversary bound (Høyer *et al.*, 2006)
- Adversary bound (Ambainis, 2000)
- Polynomial bound (Beals *et al.*, 1998)

Lower and upper bounds

- AND-OR tree evaluation (Farhi *et al.*, 2007; Ambainis *et al.*, 2007)

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

- General (negative-weight) adversary bound (Høyer *et al.*, 2006)
- Adversary bound (Ambainis, 2000)
- Polynomial bound (Beals *et al.*, 1998)

Lower and upper bounds

- AND-OR tree evaluation (Farhi *et al.*, 2007; Ambainis *et al.*, 2007)
- Span programs = Dual of general adversary bound (Reichardt *et al.*, 2009)

Quantum query complexity of
$f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$

- General (negative-weight) adversary bound (Høyer *et al.*, 2006)
- Adversary bound (Ambainis, 2000)
- Polynomial bound (Beals *et al.*, 1998)

Consequence

Quantum query complexity $=^*$ Optimum of a semidefinite program

How can this be applied?

*Up to a constant factor

Adversary Bound

Adversary Bound

$$\begin{array}{lll} \text{maximize} & \|\Gamma\| & \\ \text{subject to} & \|\Gamma \circ \Delta_j\| \le 1 & \text{for all } j \in [n]. \end{array}$$

Here: Γ is an $f^{-1}(1) \times f^{-1}(0)$-matrix with real entries, and

$$\Delta_j = \begin{cases} 1, & x_j \neq y_j; \\ 0, & \text{otherwise.} \end{cases}$$

Adversary Bound

$$\begin{array}{lll} \text{maximize} & \|\Gamma\| & \\ \text{subject to} & \|\Gamma \circ \Delta_j\| \leq 1 & \text{for all } j \in [n]. \end{array}$$

- Has been used in assumption that all entries of Γ are non-negative (original formulation).
- That has handy combinatorial variants.

Adversary Bound

$$\begin{array}{lll} \text{maximize} & \|\Gamma\| & \\ \text{subject to} & \|\Gamma \circ \Delta_j\| \le 1 & \text{for all } j \in [n]. \end{array}$$

Theorem: Suppose $X \subseteq f^{-1}(1)$, $Y \subseteq f^{-1}(0)$, and a relation $\sim$ between X and Y are such that

- for each $x \in X$, there are at least m different $y \in Y$ such that $x \sim y$;
- for each $y \in Y$, there are at least m' different $x \in X$ such that $x \sim y$;
- for each $x \in X$ and $j \in [n]$, there are at most ℓ different $y \in Y$ such that $x \sim y$ and $x_j \neq y_j$;
- for each $y \in Y$ and $j \in [n]$, there are at most ℓ' different $x \in X$ such that $x \sim y$ and $x_j \neq y_j$.

Then, any quantum algorithm computing f uses $\Omega\left(\sqrt{\frac{mm'}{\ell\ell'}}\right)$ queries.

Adversary Bound

$$\begin{aligned} &\text{maximize} && \|\Gamma\| \\ &\text{subject to} && \|\Gamma \circ \Delta_j\| \leq 1 && \text{for all } j \in [n]; \end{aligned}$$

- This special case is known to be non-tight.

Dual Adversary Bound

$$\begin{array}{lll} \text{minimize} & \max\limits_{x\in\mathcal{D}} \sum_{j\in[n]} X_j[\![x,x]\!] & \\ \text{subject to} & \sum_{j\,:\,x_j\neq y_j} X_j[\![x,y]\!] = 1 & \text{whenever } f(x)\neq f(y); \\ & X_j \succeq 0 & \text{for all } j\in[n]. \end{array}$$

Here: X_j are $\mathcal{D}\times\mathcal{D}$-matrices with real entries.

- Has almost never been used.
- An exception is formulae evaluation:
  Solving a small instance on a computer, applying tight composition results.

Learning graphs

Certificates

Definition: For a function $f\colon [m]^n \supseteq \mathcal{D} \to \{0,1\}$ and input $x \in f^{-1}(1)$, a *1-certificate* is a subset $S \subseteq [n]$ such that

$$(\forall j \in S\colon y_j = x_j) \Longrightarrow f(y) = 1 \qquad \text{for all } y \in \mathcal{D}.$$

1-certificate complexity of x is the smallest size of a 1-certificate;
1-certificate complexity of f is the maximum of 1-certificate complexities over $x \in f^{-1}(1)$.

Example: For OR function:

- any $j \in [n]$ such that $x_j = 1$ forms a 1-certificate
- 1-certificate complexity of OR is 1

Learning graphs

- Model for constructing feasible solutions for Dual Adversary bound, hence, quantum query algorithms.
  By Belovs, arXiv:1105.4024, STOC 2012.

- Randomized zero-error procedure for loading values of variables.
- For each positive input: its own procedure to load a 1-certificate.
- Complexity: from interplay between different inputs.

For OR function:
(x is a positive input, and $\{a\}$ is a 1-certificate)

I:	Load a

Transitions

I:	Load a

- Define the set of *transitions* as union over all inputs:

I:	From $\emptyset$ to $\{j\}$ for all $j \in [n]$;

Complexity

Theorem: The complexity of the learning graph is $O\left(\sum_i L_i\sqrt{T_i}\right)$ where the sum is over stages and

$$\begin{array}{rll} \text{Length} & L_i\text{:} & \text{Number of variables loaded on the stage} \\ \text{Speciality} & T_i\text{:} & \begin{pmatrix}\text{Number of transitions} \\ \text{on the stage}\end{pmatrix} / \begin{pmatrix}\text{Number of ones} \\ \text{used for one input}\end{pmatrix} \end{array}$$

	Transitions	Used	Length	Speciality
I:	From $\emptyset$ to $\{j\}$	$j = a$	1	n

Total complexity: $O(\sqrt{n})$.

Element distinctness

Formulation

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist $a \neq b$ such that $x_a = x_b$.

Formulation

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist $a \neq b$ such that $x_a = x_b$.

Complexity: $\Theta(n^{2/3})$

- Algorithm by Ambainis (2003).
  Apply quantum walk on the Johnson graph of r-subsets of $[n]$, with accepting vertices containing equal elements.
- Lower bound by Aaronson and Shi (2001).
  Using polynomial method.

Learning graphs

I:	Load r elements not from $\{a, b\}$
II:	Load a
III:	Load b

- Set of *transitions* for all inputs:

I:	From $\emptyset$ to S of r elements;
II:	From S to $S \cup \{j\}$ for $\lvert S\rvert = r$ and $j \notin S$;
III:	From S to $S \cup \{j\}$ for $\lvert S\rvert = r + 1$ and $j \notin S$.

Complexity

Length L_i: Number of variables loaded on the stage

Speciality T_i: $\begin{pmatrix}\text{Number of transitions}\\ \text{on the stage}\end{pmatrix} / \begin{pmatrix}\text{Number of ones}\\ \text{used for one input}\end{pmatrix}$

	Transitions	Used	Length	Speciality
I:	From $\emptyset$ to S of r elements	$a, b \notin S$	r	$O(1)$
II:	From S to $S \cup \{j\}$ for $\|S\| = r$ and $j \notin S$	$a, b \notin S$, $j = a$	1	$O(n)$
III:	From S to $S \cup \{j\}$ for $\|S\| = r+1$ and $j \notin S$	$a \in S$, $j = b$	1	$O(n^2/r)$

Complexity

	Transitions	Used	Length	Speciality
I:	From $\emptyset$ to S of r elements	$a, b \notin S$	r	$O(1)$
II:	From S to $S \cup \{j\}$ for $\lvert S\rvert = r$ and $j \notin S$	$a, b \notin S$, $j = a$	1	$O(n)$
III:	From S to $S \cup \{j\}$ for $\lvert S\rvert = r + 1$ and $j \notin S$	$a \in S$, $j = b$	1	$O(n^2/r)$

We get complexity:

$$O\left(\sum_i L_i \sqrt{T_i}\right) = O(r + \sqrt{n} + n/\sqrt{r}) = O(n^{2/3})$$

when $r = n^{2/3}$.

Where does a man hide a leaf? In the forest.
But what does he do if there is no forest?..
He grows a forest to hide it in.

Gilbert Keith Chesterton

Hiding

Naïve learning graph

I:	Load a
II:	Load b

Complexity: $O(n)$

Optimal learning graph

I:	Load r elements not from $\{a, b\}$
II:	Load a
III:	Load b

Complexity: $O(n^{2/3})$.

- Before loading b, a is *hidden* among the r loaded elements.

More generality

A similar algorithm solves *any* problem with 1-certificate complexity $k = O(1)$:

I:	Load r elements not from $\{a_1, a_2, \dots, a_k\}$
II.1:	Load a_1
	$\vdots$
II.k:	Load a_k

- Complexity is $O(n^{k/(k+1)})$.
- Corresponds to quantum walk on the Johnson graph.

Lower bound

The k-sum problem:

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist pairwise distinct $a_1, \ldots, a_k$ such that $x_{a_1} + x_{a_2} + \cdots + x_{a_k}$ is divisible by m.

- Belovs and Špalek, arXiv:1206.6528

  Lower bound of $\Omega(n^{k/(k+1)})$ using adversary method

- Hence, quantum walk on the Johnson graph is optimal for this problem.
- Due to *certificate complexity barrier* no adversary with non-negative entries exist with bound $\omega(\sqrt{n})$.
- This lower bound has applications in quantum Merkle puzzles.

Need for Structure

The k-sum problem:

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist pairwise distinct $a_1, \ldots, a_k$ such that $x_{a_1} + x_{a_2} + \cdots + x_{a_k}$ is divisible by m.

- Given a $(k-1)$-tuple of input variables, we have absolutely no idea whether they form a part of a 1-certificate.

Need for Structure

The k-sum problem:

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist pairwise distinct $a_1, \ldots, a_k$ such that $x_{a_1} + x_{a_2} + \cdots + x_{a_k}$ is divisible by m.

- Given a $(k-1)$-tuple of input variables, we have absolutely no idea whether they form a part of a 1-certificate.
- There is no structure between the values of different variables.

Need for Structure

The k-sum problem:

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist pairwise distinct $a_1, \ldots, a_k$ such that $x_{a_1} + x_{a_2} + \cdots + x_{a_k}$ is divisible by m.

- Given a $(k-1)$-tuple of input variables, we have absolutely no idea whether they form a part of a 1-certificate.
- There is no structure between the values of different variables.

What happens if we introduce structure?

Various Problems

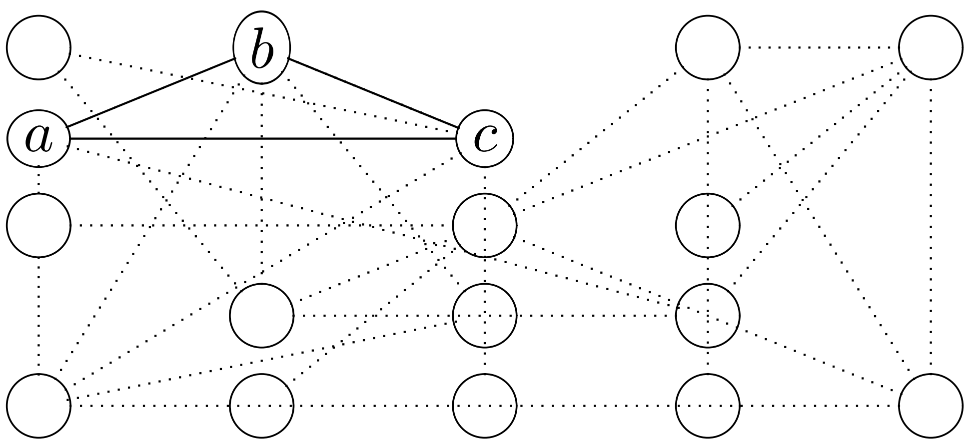
Various Problems:
Triangle Detection

Triangle Problem

Given $x_{i,j} \in \{0,1\}$, with $1 \le i < j \le n$, detect whether there exist $1 \le a < b < c \le n$ such that

$$x_{a,b} = x_{a,c} = x_{b,c} = 1.$$

NB: the number of input variables is $\Theta(n^2)$.



Triangle Problem

Given $x_{i,j} \in \{0,1\}$, with $1 \leq i < j \leq n$, detect whether there exist $1 \leq a < b < c \leq n$ such that

$$x_{a,b} = x_{a,c} = x_{b,c} = 1.$$

- There is structure between the variables.
- Quantum walk on the Johnson graph would give: $O(n^{3/2})$.

Triangle Problem

Given $x_{i,j} \in \{0,1\}$, with $1 \leq i < j \leq n$, detect whether there exist $1 \leq a < b < c \leq n$ such that

$$x_{a,b} = x_{a,c} = x_{b,c} = 1.$$

- $O(n^{13/10})$ query algorithm by Magniez, Santha, Szegedy (2005) using two quantum walks on the Johnson graph: one inside another.
- $\Omega(n)$ lower bound, trivial by reduction to Grover.

Triangle Problem

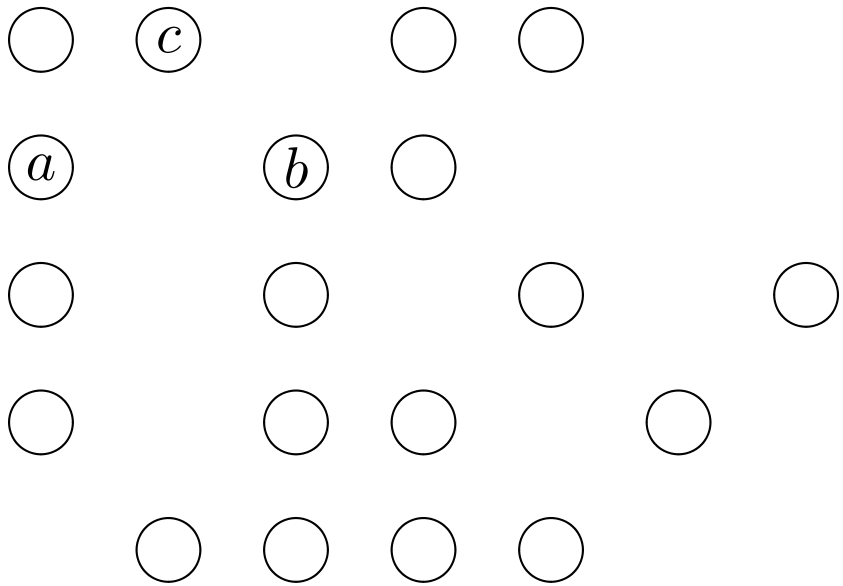
Given $x_{i,j} \in \{0, 1\}$, with $1 \leq i < j \leq n$, detect whether there exist $1 \leq a < b < c \leq n$ such that

$$x_{a,b} = x_{a,c} = x_{b,c} = 1.$$

- $O(n^{13/10})$ query algorithm by Magniez, Santha, Szegedy (2005) using two quantum walks on the Johnson graph: one inside another.
- $\Omega(n)$ lower bound, trivial by reduction to Grover.

- $O(n^{35/27})$ query algorithm by Belovs, arXiv:1105.4024, STOC 2012.
- $O(n^{9/7})$ query algorithm by Lee, Magniez and Santha, to appear in SODA 2013.
- Can be generalized to other subgraph containment problems.

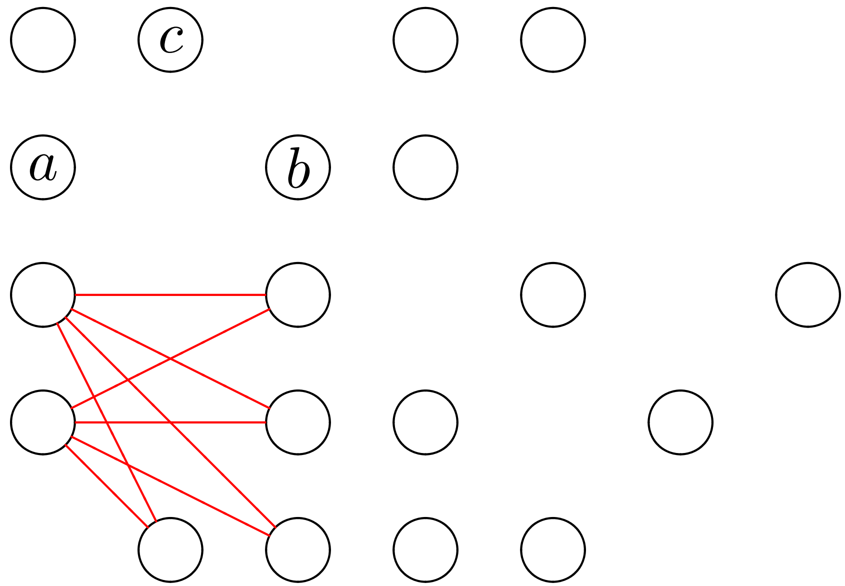
Learning graph



In the beginning nothing is loaded.

Continue as follows...

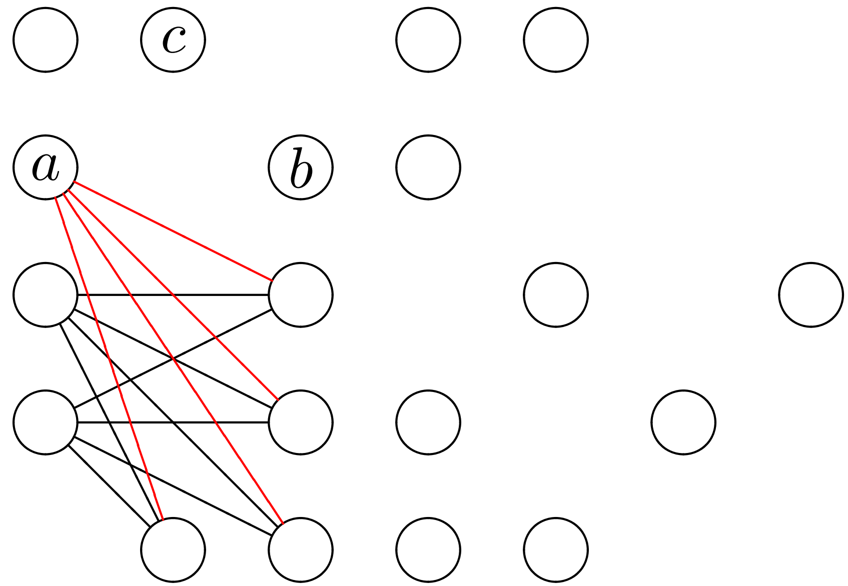
Learning graph



I: Take disjoint $A, B \subseteq [n] \setminus \{a, b, c\}$ of sizes $n^{4/7}$ and $n^{5/7}$, and load all edges between A and B

Length:	$\|A\|\|B\| = n^{9/7}$
Speciality:	1
Complexity:	$n^{9/7}$

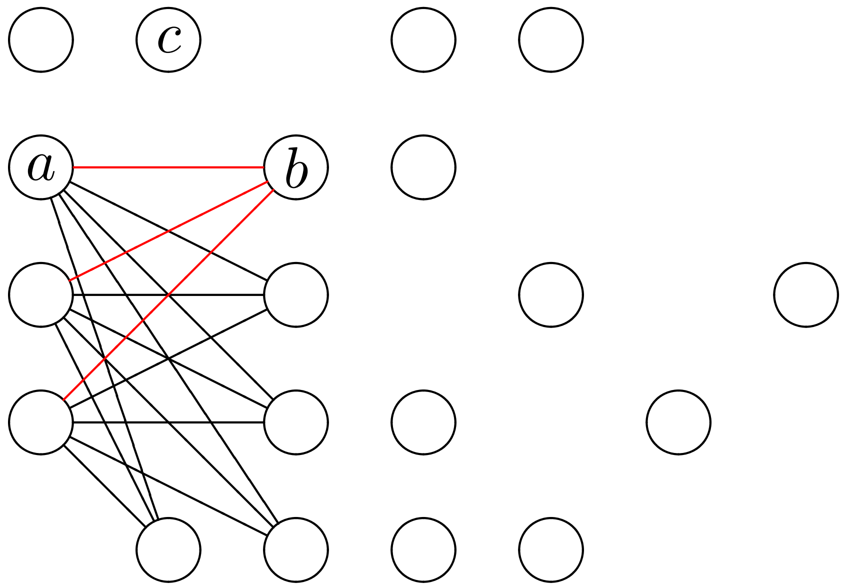
Learning graph



II: Add a to A and load all edges between a and B

Length: $|B| = n^{5/7}$
Speciality: n
Complexity: $n^{17/14}$

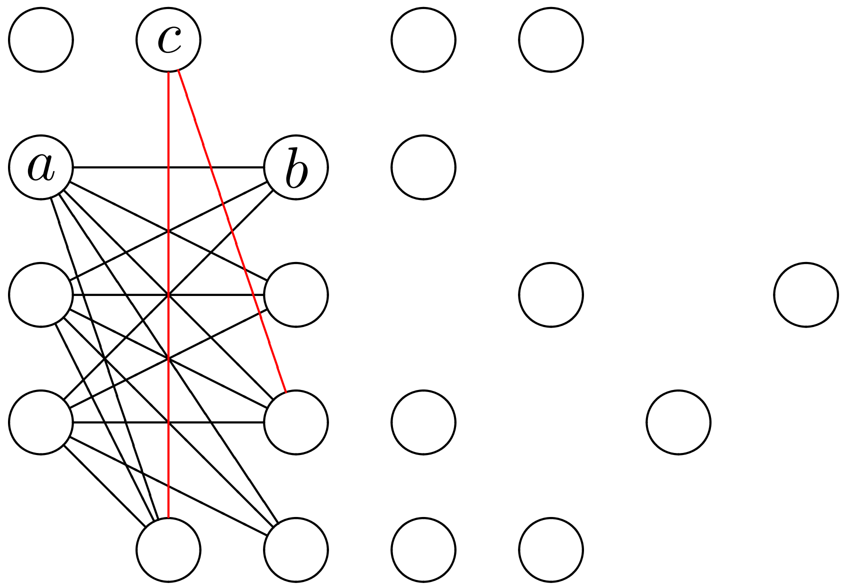
Learning graph



III: Add b to B and load all edges between b and A

Length: $|A| = n^{4/7}$
Speciality: $n^2/|A| = n^{10/7}$
Complexity: $n^{9/7}$

Learning graph



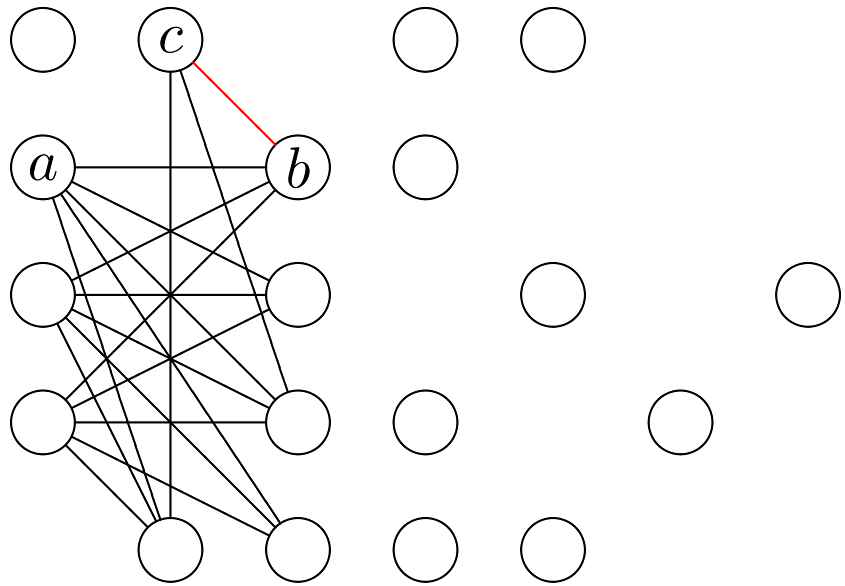
IV: Load $\ell = n^{3/7}$ edges connecting c to vertices in B, but b

Length: $\ell = n^{3/7}$

Speciality: $n^3/(|A||B|) = n^3/(n^{4/7}n^{5/7}) = n^{12/7}$

Complexity: $n^{9/7}$

Learning graph

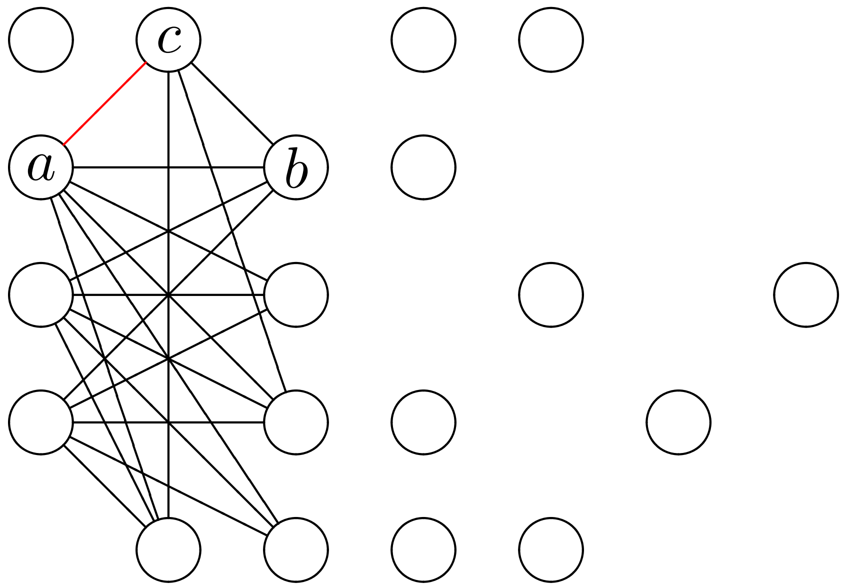


V: Load edge bc

Length: 1

Speciality: $n^3/|A| = n^3/n^{4/7} = n^{17/7}$

Complexity: $n^{17/14}$



VI: Load edge ac

Length:	1
Speciality:	$n^3/\ell = n^{18/7}$
Complexity:	$n^{9/7}$

Overall learning graph

I:	Take disjoint $A, B \subseteq [n] \setminus \{a, b, c\}$ of sizes $n^{4/7}$ and $n^{5/7}$ and load all edges between A and B
II:	Add a to A and load all edges between a and B
III:	Add b to B and load all edges between b and A
IV:	Load $\ell = n^{3/7}$ edges connecting c to elements in B, but b
V:	Load edge bc
VI:	Load edge ac

Stage	I	II	III	IV	V	VI
Length	$n^{9/7}$	$n^{5/7}$	$n^{4/7}$	$n^{3/7}$	1	1
Speciality	1	n	$n^{10/7}$	$n^{12/7}$	$n^{17/7}$	$n^{18/7}$
Complexity	$n^{9/7}$	$n^{17/14}$	$n^{9/7}$	$n^{9/7}$	$n^{17/14}$	$n^{9/7}$

Total complexity: $O(n^{9/7})$.

Various Problems:
k-distinctness

k-distinctness

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist $a_1, \ldots, a_k$, all distinct, such that

$$x_{a_1} = x_{a_2} = \cdots = x_{a_k}.$$

(If $k = 2$, this is element distinctness.)

- This time: there is structure between *the values* of the variables.

k-distinctness

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist $a_1, \ldots, a_k$, all distinct, such that

$$x_{a_1} = x_{a_2} = \cdots = x_{a_k}.$$

(If $k = 2$, this is element distinctness.)

- $O(n^{k/(k+1)})$ algorithm by Ambainis (2003)
  Using quantum walk on the Johnson graph
- $\Omega(n^{2/3})$ lower bound, by reduction to the Element Distinctness problem.

k-distinctness

Given $x_1, \ldots, x_n \in [m]$, detect whether there exist $a_1, \ldots, a_k$, all distinct, such that

$$x_{a_1} = x_{a_2} = \cdots = x_{a_k}.$$

(If $k = 2$, this is element distinctness.)

- $O(n^{k/(k+1)})$ algorithm by Ambainis (2003)
  Using quantum walk on the Johnson graph
- $\Omega(n^{2/3})$ lower bound, by reduction to the Element Distinctness problem.

- $O\left(n^{1-2^{k-2}/(2^k-1)}\right) = o(n^{3/4})$ query algorithm by Belovs,
  using more complex learning graphs
  arXiv:1205.1534, to appear in FOCS 2012.

Previous Approach

Consider 3-distinctness:
The task is to load a triple $\{a, b, c\}$ of equal elements.

I:	Load r elements not from $\{a, b, c\}$
II.1:	Load a
II.2:	Load b
II.3:	Load c

On step II.3, while loading c, a and b are hidden in the set S of loaded variables:

$$S: 1\ 9\ 7\ 8\ 5\ 6\ 2\ 3\ 7\ 5\ 4\ 0\ 0\ 6$$

Some modifications

In the previous algorithm, while loading c, a and b were hidden in S:

$$S\colon \mathtt{1\ 9\ 7\ 8\ 5\ 6\ 2\ 3\ 7\ 5\ 4\ 0\ 0\ 6}$$

We divide S into S_1 and S_2, for a and b, respectively:

$$\overbrace{\mathtt{1\ 9\ 7\ 8\ 5\ 6\ 2}}^{S_1}\ \overbrace{\mathtt{3\ 7\ 5\ 4\ 0\ 0\ 6}}^{S_2}$$

Some elements of S_2 just can't be b. Their values are irrelevant.

$$\overbrace{\mathtt{1\ 9\ 7\ 8\ 5\ 6\ 2}}^{S_1}\ \overbrace{\mathtt{*\ 7\ 5\ *\ *\ *\ 6}}^{S_2}$$

Some modifications

Some elements of S_2 just can't be b. Their values are irrelevant.

$$\overbrace{\mathtt{1\ 9\ 7\ 8\ 5\ 6\ 2}}^{S_1}\ \overbrace{\mathtt{*\ 7\ 5\ *\ *\ *\ 6}}^{S_2}$$

The value of n boolean variables can be learned faster than in n queries, if there is a bias between the number of zeros and ones.
Elements having a pair in S_1 are much rarer than the ones that don't. S_2 can be enlarged.

$$\overbrace{\mathtt{1\ 9\ 7\ 8\ 5\ 6\ 2}}^{S_1}\ \overbrace{\mathtt{*\ *\ *\ *\ 7\ *\ *\ 5\ *\ *\ *\ 6\ *\ *\ *\ *\ 1\ *\ *\ *\ \ldots}}^{S_2}$$

Learning graph

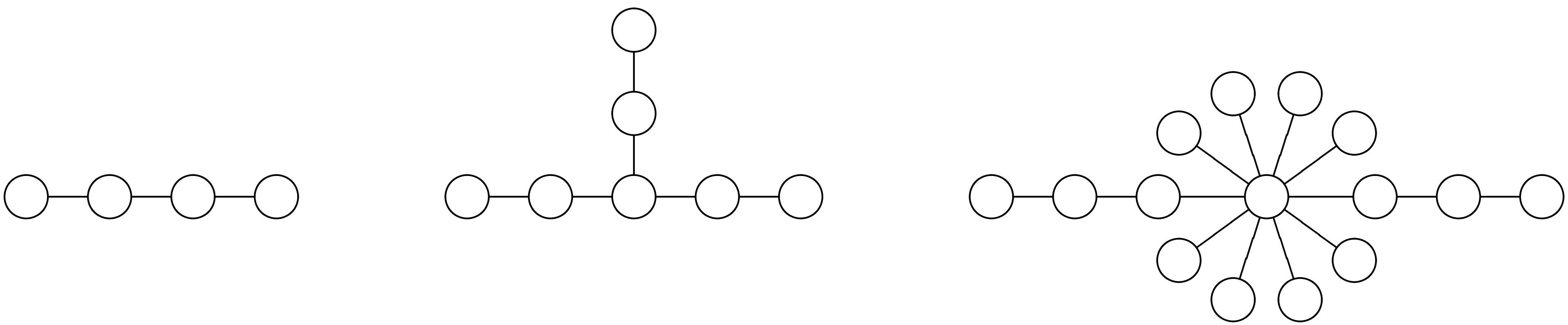
Let $M = \{a_1, \ldots, a_k\}$ be the set of equal elements.

I.1	Load a set S_1 of r_1 elements not from M.
I.2	Load a set S_2 of r_2 elements not from M, uncovering those elements only that have a match in S_1.
I.3	Load a set S_3 of r_3 elements not from M, uncovering those elements only that have a match among the uncovered elements of S_2.
	$\vdots$
I.$(k-1)$	Load a set S_{k-1} of r_{k-1} elements not from M, uncovering those elements only that have a match among the uncovered elements of S_{k-2}.
II.1	Load a_1 and add it to S_1.
	$\vdots$
II.$(k-1)$	Load a_{k-1} and add it to S_{k-1}.
II.k	Load a_k.

Even more subgraph containment

Belovs and Reichardt (arXiv:1203.2603, ESA 2012), inspired by the paper by A. Childs and R. Kothari.

Any fixed graph of the three following types can be detected in $O(n)$ quantum queries:

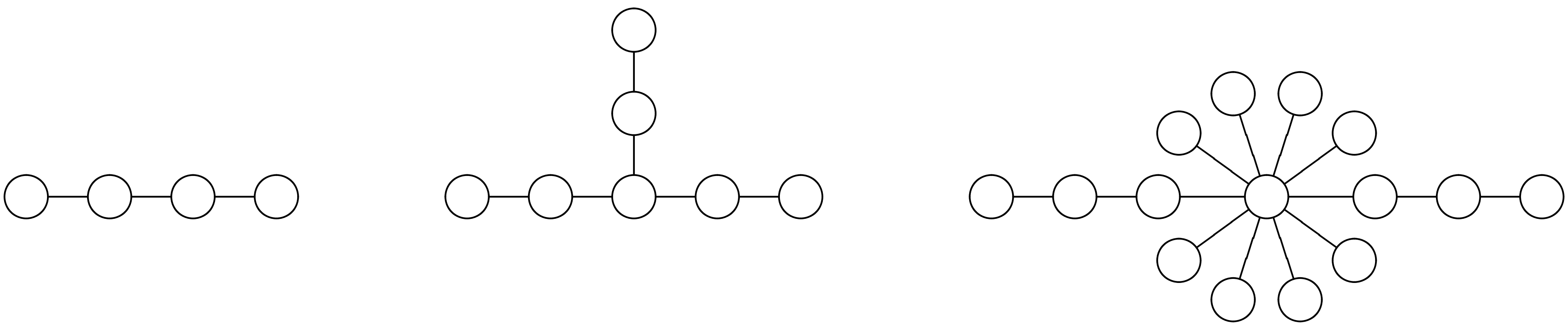


NB: Improvement from almost $O(n^{3/2})$ for large subgraphs.

Even more subgraph containment

Belovs and Reichardt (arXiv:1203.2603, ESA 2012), inspired by the paper by A. Childs and R. Kothari.

Any fixed graph of the three following types can be detected in $O(n)$ quantum queries:



NB: Improvement from almost $O(n^{3/2})$ for large subgraphs.

- the algorithm can be implemented in $\tilde{O}(n)$ time and $O(\log n)$ space.

Summary

- Learning graphs reduce to Adversary Upper Bound
- They can mimic quantum walks on the Johnson graph
- Learning graphs allow more control compared to previous quantum walks used to construct algorithms
- Analysis is combinatorial
- No spectral analysis is required

Open Problems

- More adversary upper and lower bounds!
  - Lower bound for collision and set equality problems
  - Lower bound for k-distinctness. Is the algorithm tight?
- Time-efficient implementation of other learning graphs.
  - k-distinctness is more likely

Thank you!