Title: Symmetries, graph properties, and quantum speedups

Speakers: Supartha Podder

Series: Perimeter Institute Quantum Discussions

Date: December 09, 2020 - 4:00 PM

URL: http://pirsa.org/20120020

Abstract: Aaronson and Ambainis (2009) and Chailloux (2018) showed that fully symmetric (partial) functions do not admit exponential quantum query speedups. This raises a natural question: how symmetric must a function be before it cannot exhibit a large quantum speedup? In this work, we prove that hypergraph symmetries in the adjacency matrix model allow at most a polynomial separation between randomized and quantum query complexities. We also show that, remarkably, permutation groups constructed out of these symmetries are essentially the only permutation groups that prevent super-polynomial quantum speedups. We prove this by fully characterizing the primitive permutation groups that allow super-polynomial quantum speedups. In contrast, in the adjacency list model for bounded-degree graphs (where graph symmetry is manifested differently), we exhibit a property testing problem that shows an exponential quantum speedup. These results resolve open questions posed by Ambainis, Childs, and Liu (2010) and Montanaro and de Wolf (2013). Based on: arxiv:2006.12760

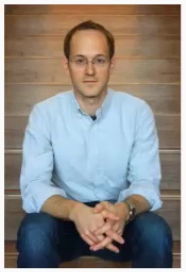
Symmetries, graph properties, and quantum speedups

Full version: arXiv:2006.12760

Supartha Podder

University of Ottawa

Shalev Ben-David
University of Waterloo



Andrew M. Childs
University of Maryland

András Gilyén
Caltech/Berkeley

William Kretschmer
UT Austin

Daochen Wang
University of Maryland

The power of quantum computers

Quantum computers can sometimes solve problems significantly faster than classical computers.

Some problems for which quantum computation exhibits exponential speedups

Quantum simulations, period finding, factoring, discrete logarithm problem, quantum linear algebra, Jones polynomial approximation etc.

While for others it does not

Unstructured search, collision problem, formula evaluation, st-connectivity, finding subgraphs, subgraph isomorphism problems, certain computational geometry problems, convex optimizations etc.

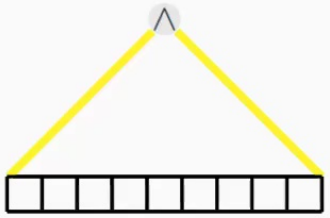
Query complexity model

Ignore the processing time: Only consider the number of times input is read:

- Given a known function $f : \{0,1\}^n \to \{0,1\}$ and
- Given oracle access to the bits of an unknown string $x \in \{0,1\}^n$, compute $f(x)$.

$$\textit{Oracle } O : i \to x_i \textit{ for any } i \in [n].$$

- Cost: Minimum number of bits needed to be queried.
  Example: $AND : \{0,1\}^n \to \{0,1\}, AND(x) = 1$ iff $\forall i \in [n], x_i = 1$.



AND function

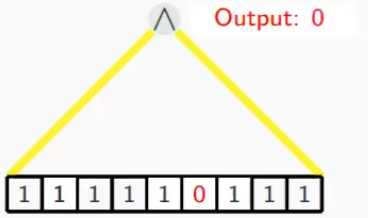
Query complexity model

Ignore the processing time: Only consider the number of times input is read:

- Given a known function $f : \{0,1\}^n \to \{0,1\}$ and
- Given oracle access to the bits of an unknown string $x \in \{0,1\}^n$, compute $f(x)$.

Oracle $O : i \to x_i$ for any $i \in [n]$.

- Cost: Minimum number of bits needed to be queried.
  Example: $AND : \{0,1\}^n \to \{0,1\}$, $AND(x) = 1$ iff $\forall i \in [n], x_i = 1$.



AND function

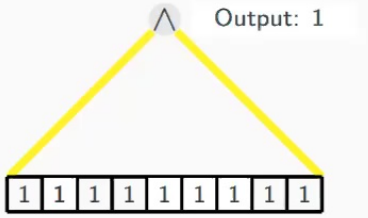
Query complexity model

Ignore the processing time: Only consider the number of times input is read:

- Given a known function $f : \{0,1\}^n \to \{0,1\}$ and
- Given oracle access to the bits of an unknown string $x \in \{0,1\}^n$, compute $f(x)$.

$$\textit{Oracle } O : i \to x_i \textit{ for any } i \in [n].$$

- Cost: Minimum number of bits needed to be queried.
  Example: $AND : \{0,1\}^n \to \{0,1\}$, $AND(x) = 1$ iff $\forall i \in [n], x_i = 1$.



AND function
D(*AND*)=n

- The deterministic (or randomized/quantum) complexity of f is the number of queries needed on the worst input x by the best deterministic (or randomized/quantum) algorithm computing f.
- For all f, $Q(f) \leq R(f) \leq D(f) \leq n$.

No super-polynomial speedups for total functions

Beals, Buhrman, Cleve, Mosca, de Wolf (1998) showed that for all total functions classical and quantum query complexity are polynomially (power 6) related.

Aaronson, Ben-David, Kothari, Tal 2020

For all total functions classical and quantum query complexity are at most power 4 related.

Relations between quantum and classical computation

No super-polynomial speedups for total functions
Beals, Buhrman, Cleve, Mosca, de Wolf (1998) showed that for all total functions classical and quantum query complexity are polynomially (power 6) related.

Aaronson, Ben-David, Kothari, Tal 2020
For all total functions classical and quantum query complexity are at most power 4 related.

So promises are required for super polynomial speedups.

- $f : P \to \{0,1\}$.
- If $P = \{0,1\}^n$, then function is total.
- Otherwise if $P \subset \{0,1\}^n$, the function is a partial function.

Partial functions and speedup

Do all promised problems give super polynomial speedup?

x_1	x_2	x_3	x_4	AND
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Example: 2-to-1 collision problem

Given black-box queries to the input $[m]^n$ decide whether a sequence of numbers $(x_1, \cdots, x_n) \in [m]^n$ is one-to-one (each number appears once) or two-to-one (each number appears twice).

- $R(f) = \Theta(n^{1/2}), Q(f) = \Theta(n^{1/3})$.

Example: Simon's problem

$f : \{0,1\}^n \to \{0,1\}^n$ is defined as: $f(x) = f(y) \iff (x \oplus y) \in \{0^n, s \neq 0^n\}$. Find out if $s \neq 0^n$.

- $R(f) = \Theta(n^{1/2}), Q(f) = \Theta(\log n)$.

Symmetric Functions

In this direction, it is known that we cannot have super polynomial speedups if the function is too symmetric.

Partial functions and speedup

Do all promised problems give super polynomial speedup?

x_1	x_2	x_3	x_4	AND
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Example: 2-to-1 collision problem

Given black-box queries to the input $[m]^n$ decide whether a sequence of numbers $(x_1, \cdots, x_n) \in [m]^n$ is one-to-one (each number appears once) or two-to-one (each number appears twice).

- $R(f) = \Theta(n^{1/2}), Q(f) = \Theta(n^{1/3})$.

Example: Simon's problem

$f : \{0,1\}^n \to \{0,1\}^n$ is defined as: $f(x) = f(y) \iff (x \oplus y) \in \{0^n, s \neq 0^n\}$. Find out if $s \neq 0^n$.

- $R(f) = \Theta(n^{1/2}), Q(f) = \Theta(\log n)$.

Need of structure for exponential speedups:

Structured basically means that we are trying to determine some property of the input, assuming that the input satisfies some global regularity.

Symmetric Functions

In this direction, it is known that we cannot have super polynomial speedups if the function is too symmetric.

Permutation invariant functions
A partial function $f : P \to \{0,1\}$ is permutation invariant if for all permutation $\pi \in S_n$:

- $x = (x_1, x_2, \cdots, x_n) \in P \implies x \circ \pi = (x_{\pi_1}, \cdots, x_{\pi_n}) \in P$ and
- $f(x) = f(x \circ \pi)$.

No exponential speedups for symmetric functions (Ambainis-Aaronson 2014, Chailloux 2018)

For all permutation invariant partial functions f,

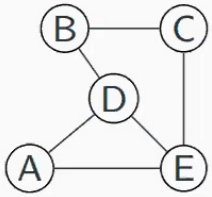
$$R(f) \leq Q(f)^3.$$

Other Symmetric functions?

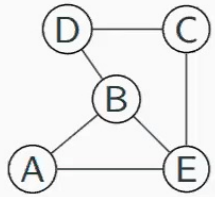
Question:
What about other kinds of symmetry?

Symmetric under group action G
Let $f : P \to \{0,1\}$ be a function and $G \subseteq S_n$ be a permutation group. We say f is symmetric under G for every $\pi \in G$:

- $x = (x_1, x_2, \cdots, x_n) \in P \implies x \circ \pi = (x_{\pi_1}, \cdots, x_{\pi_n}) \in P$ and
- $f(x) = f(x \circ \pi)$.



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$



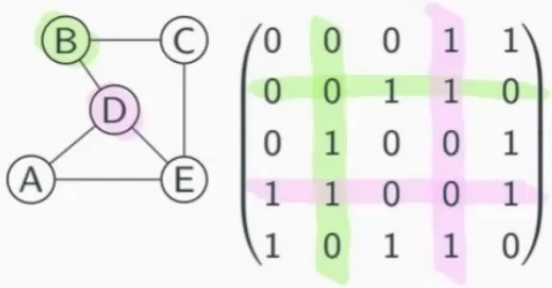
$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Graph properties: less symmetry

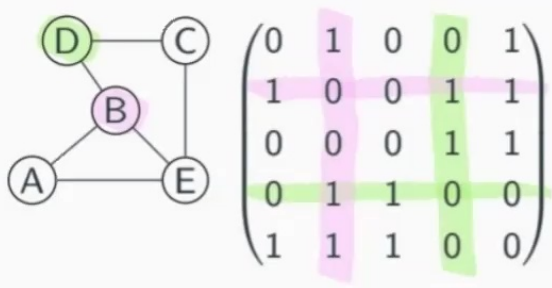
- Graph properties are invariant under vertex renaming, i.e., property depends only on the isomorphism class.

7

Graph properties



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$



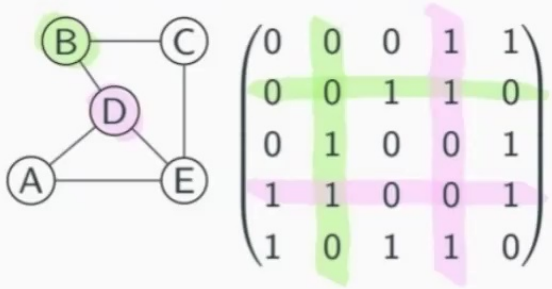
$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Graph properties: less symmetry

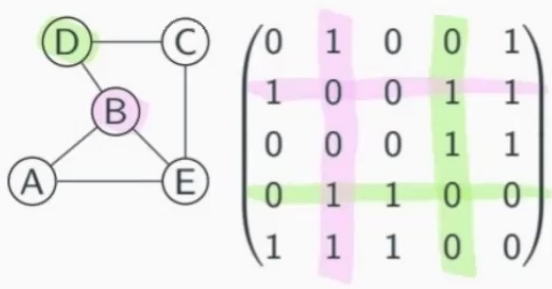
- Graph properties are invariant under vertex renaming, i.e., property depends only on the isomorphism class.
- A graph $G \in P$, then for any $\pi \in S_n$, $\pi(G) \in P$ and $f(G) = f(\pi(G))$.
- An n-vertex graph is given as an oracle access to its edges (in the adjacency matrix representation).

$$O : (i, j) \to A_{ij}.$$

Graph properties



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$



$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

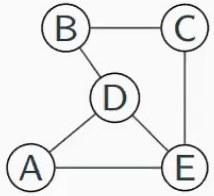
Graph properties: less symmetry

- Graph properties are invariant under vertex renaming, i.e., property depends only on the isomorphism class.
- A graph $G \in P$, then for any $\pi \in S_n$, $\pi(G) \in P$ and $f(G) = f(\pi(G))$.
- An n-vertex graph is given as an oracle access to its edges (in the adjacency matrix representation).

$$O : (i, j) \to A_{ij}.$$

- Vertex permutation induces edge permutation implies much less symmetry: $(n! \ll (n^2)!)$.

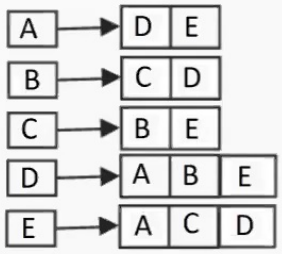
Graph properties



Graph

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Adjacency matrix



Adjacency list

Question: (Ambainis, Childs, and Liu 2010 and Montanaro and de Wolf 2013)

Can we show that exponential speedups are not possible for any graph properties (in both adjacency matrix and adjacency list model)?

In fact, they asked the question about property testing problems.

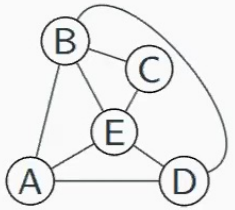
Our results: Graph properties

No super-polynomial speedups is possible for graph properties in the adjacency matrix model.
For all k-uniform hypergraph properties f:

$$R(f) = O(Q(f)^{3k}).$$

Corollary
For all graph properties f in the adjacency matrix model:

$$R(f) = O(Q(f)^{6}).$$



Same holds for directed graph symmetries, bipartite graph symmetries etc.

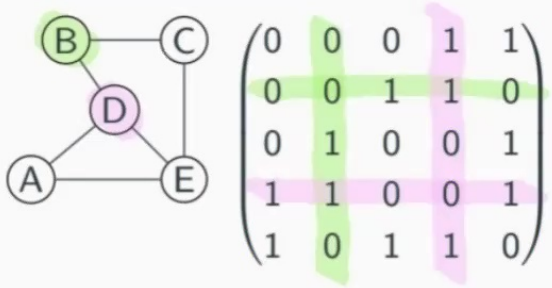
Our results - graph properties in the adjacency list model

Exponential speedups is possible for a property testing problem in adjacency list model

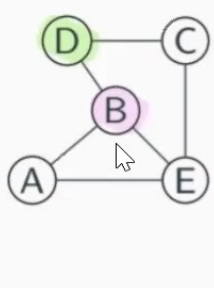
There is a graph *property testing* problem in the adjacency list model that can be solved with $poly(n)$ quantum queries, but requires $2^{\Omega(n)}$ classical queries.

Hence, for graph properties whether or not we can get large quantum speedups depends on what model the graph is represented in.

Graph properties



$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$



$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Graph properties: less symmetry

- Graph properties are invariant under vertex renaming, i.e., property depends only on the isomorphism class.
- A graph $G \in P$, then for any $\pi \in S_n$, $\pi(G) \in P$ and $f(G) = f(\pi(G))$.
- An n-vertex graph is given as an oracle access to its edges (in the adjacency matrix representation).

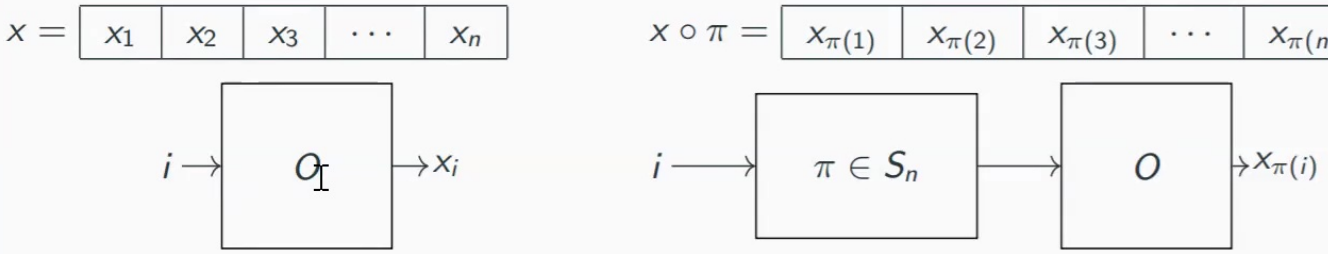
$$O : (i, j) \to A_{ij}.$$

Our results - other types of symmetries

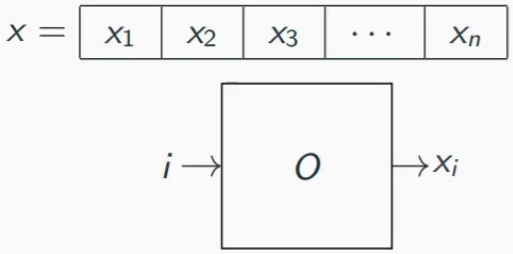
Question:
Can we classify all permutation groups G such that functions f symmetric under G have $R(f) = O(Q(f)^{O(1)})$?

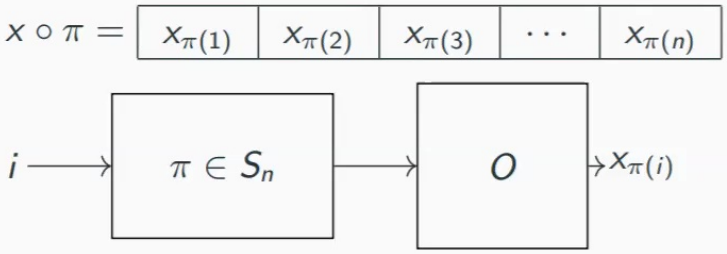
No super-polynomial speedups for graph symmetries

Chailloux's proof and extension to graph symmetries

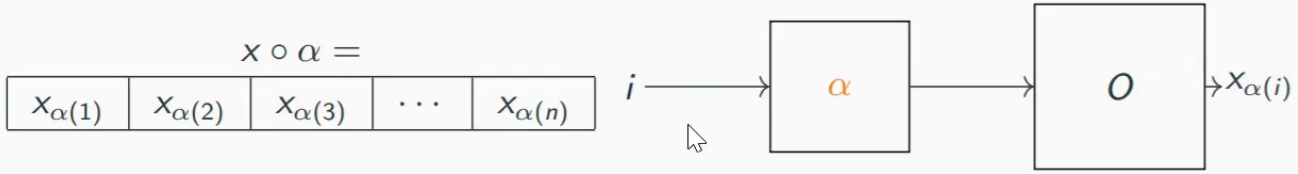


Chailloux's proof and extension to graph symmetries





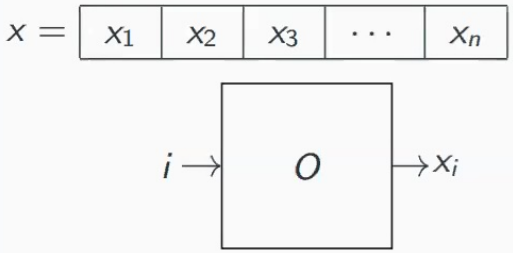
Consider a function with small range, $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$.

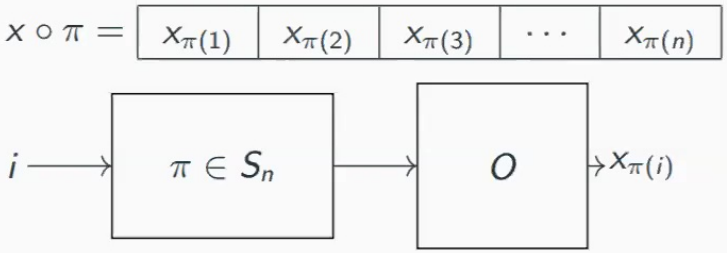


Theorem (Zhandry 2015)

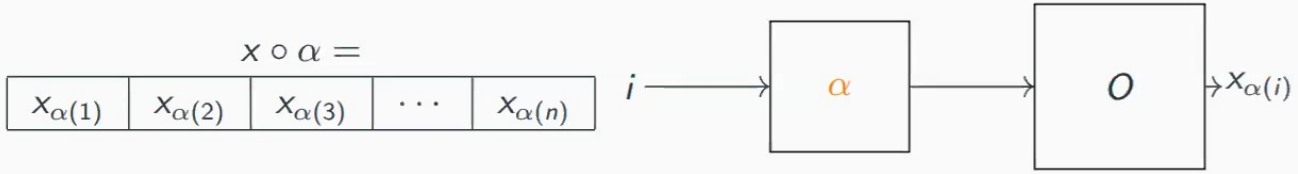
Distinguishing a random small range function $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$ from a random permutation $\pi \in S_n$ requires $\Omega(r^{1/3})$ quantum queries.

Chailloux's proof and extension to graph symmetries





Consider a function with small range, $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$.

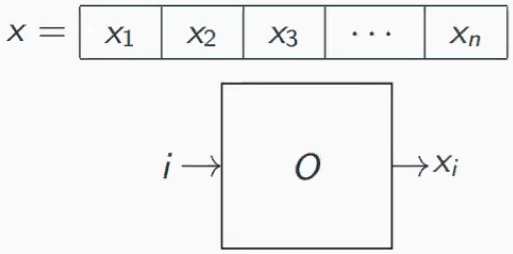


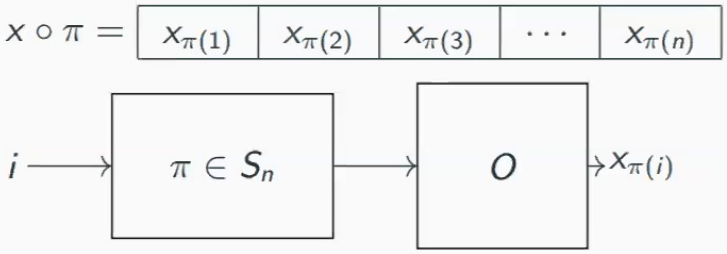
Theorem (Zhandry 2015)

Distinguishing a random small range function $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$ from a random permutation $\pi \in S_n$ requires $\Omega(r^{1/3})$ quantum queries.

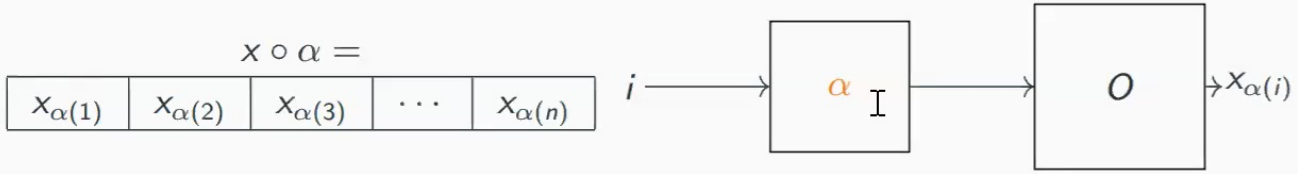
- Thus if $r \approx Q(f)^3$, then the quantum algo does not distinguish $x \circ \pi$ from $x \circ \alpha$.

Chailloux's proof and extension to graph symmetries





Consider a function with small range, $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$.



Theorem (Zhandry 2015)

Distinguishing a random small range function $\alpha : [n] \to [n]$ with $|\alpha([n])| = r$ from a random permutation $\pi \in S_n$ requires $\Omega(r^{1/3})$ quantum queries.

- Thus if $r \approx Q(f)^3$, then the quantum algo does not distinguish $x \circ \pi$ from $x \circ \alpha$.
- Quantum algorithm on $x \circ \alpha$ can be simulated with r classical queries.

Graph symmetries

- More generally, if distinguishing a random function with range r from a random permutation $\pi \in G$ requires $\Omega(r^{1/c})$ quantum queries, then $Q(f) = O(R(f)^c)$ for every f symmetric under G.
- We show this by reductions to S_n.
- Furthermore, we show that a version of Zhandry's result that is sufficient for our purposes follows easily from the collision lower bound.

For $n, r \in \mathbb{N}$, let $D_{n,r}$ be the set of all strings in $[n]^n$ such that there are at most r unique alphabets.

Theorem: Distinguishing S_n from a small range function:
There is a universal constant c such that any quantum algorithm distinguishing a permutation in S_n from a string in $D_{n,r}$ must make at least $r^{1/3}/c$ queries.

Graph symmetries

- More generally, if distinguishing a random function with range r from a random permutation $\pi \in G$ requires $\Omega(r^{1/c})$ quantum queries, then $Q(f) = O(R(f)^c)$ for every f symmetric under G.
- We show this by reductions to S_n.
- Furthermore, we show that a version of Zhandry's result that is sufficient for our purposes follows easily from the collision lower bound.

For $n, r \in \mathbb{N}$, let $D_{n,r}$ be the set of all strings in $[n]^n$ such that there are at most r unique alphabets.

Theorem: Distinguishing S_n from a small range function:
There is a universal constant c such that any quantum algorithm distinguishing a permutation in S_n from a string in $D_{n,r}$ must make at least $r^{1/3}/c$ queries.

Proof.
(by reduction from the collision bound)

- When n is multiple of r: each (n/r)-to-1 function has range r.
  - Distinguishing (n/r)-to-1 from 1-to-1 functions is a sub-problem of distinguishing $D_{n,r}$ from S_n.
  - From collision lower bound $\Omega(r^{1/3})$ lower bound follows.
- When n is not a multiple of r: set $n' = r\lceil n/r \rceil$.

Well-shuffling

Definition: Well-shuffling permutation groups

We say $\mathcal{G}$ is well-shuffling *with power* $a > 0$ if there exists $b > 0$ such that any quantum query complexity of distinguishing a permutation from G from a string in $D_{n,r}$ (denoted $cost(G, r)$) requires at least $r^{1/a}/b$ queries for all $G \in \mathcal{G}$ and all $r \in \mathbb{N}$.

Well-shuffling

Definition: Well-shuffling permutation groups
We say $\mathcal{G}$ is well-shuffling *with power* $a > 0$ if there exists $b > 0$ such that any quantum query complexity of distinguishing a permutation from G from a string in $D_{n,r}$ (denoted $cost(G, r)$) requires at least $r^{1/a}/b$ queries for all $G \in \mathcal{G}$ and all $r \in \mathbb{N}$.

Theorem: No super-polynomial speedups for well-shuffling permutation groups:
If $\mathcal{G}$ is a well-shuffling collection of permutation groups with power a, then for all $f \in F(\mathcal{G})$ we have $R(f) = O(Q(f)^a)$.

The set of symmetric group $\mathcal{S} = \{S_n\}_{n\in\mathbb{N}}$ is well-shuffling with power 3.

Directed graph symmetries: reductions to full symmetries

Extension: G acting on $[n]$ to H acting on $[n]^\ell$.

- G acting on D, $\ell \in \mathbb{N}$: $G^{(\ell)}$ acting on D^ℓ as $\pi(i_1, i_2, \cdots, i_\ell) = (\pi(i_1), \pi(i_2), \cdots, \pi(i_\ell))$ for each $\pi \in G$.
- $G^{\langle \ell \rangle}$ is $G^{(\ell)}$ with domain $D^{\langle \ell \rangle} \subseteq D^{(\ell)}$ (distinct ℓ-tuples).
- $H = G^{(\ell)}$: $cost(H, r^\ell) \geq cost(G, r)/\ell$.
- $H = G^{\langle \ell \rangle}$: $cost(H, r^\ell) \geq cost(G, r)/\ell$.

Directed graph symmetries $G_k = S_k^{\langle 2 \rangle}$: $cost(G_k, r) \geq cost(S_k, \sqrt{r})/2$.

Corollary: The set of all directed graph symmetries is well-shuffling with power 6.

A quick pit stop: Block system

Block system

Let G be a permutation group on domain D. A partition $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ of D is called a *block system* for G if $\mathcal{B}$ is G-invariant. Formally, this means that $\pi(B_i) \in \mathcal{B}$ for every $\pi \in G$ and $i \in [k]$, where $\pi(B_i) := \{\pi(x) : x \in B_i\}$. The sets B_i are called *blocks* for G.

Example:

Consider the action of the group of linear transformations on the nonzero elements of a vector space. The set of lines through the origin (i.e., nonzero scalar multiples of a nonzero vector) form a block system.

Graph symmetries: reductions to full symmetries

Graph symmetries: G_k acting on $[n]$ with $n = k(k-1)/2$ ($[n]$ represents the set of all possible edges in a k-vertex graph)

Directed to undirected graph symmetries
Let $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ is a equal size block system for a permutation group G. Let H be the permutation group on $[k]$ induced by the action of G on the blocks. Then $cost(H, r) \geq cost(G, r)$.

The set of graph symmetries is well-shuffling with power 6

Proof.

- Let G be a directed graph symmetries on domain $k(k-1)$.
- Partition this into $k(k-1)/2$ sets of size 2: $\{(x, y), (y, x)\}$ for $x, y \in [k]$.
- The induced permutation group H is preciously the undirected graph symmetry on graph of size k.
- $cost(H, r) \geq cost(G, r)$.

□

Stabilizer and base

Stabilizer

Let G be a permutation group on domain D. The *pointwise stabilizer* of a set $S \subseteq D$ is the subgroup

$$\mathrm{Stab}_G(S) := \{\pi \in G : \pi(i) = i \ \forall i \in S\}.$$

The *setwise stabilizer* of a set $S \subseteq D$ is the subgroup

$$\mathrm{Stab}_G(\{S\}) := \{\pi \in G : \pi(i) \in S \ \forall i \in S\}.$$

Base

Let G be a permutation group on domain D. A *base* for G is a set $S \subseteq D$ such that the pointwise stabilizer $\mathrm{Stab}_G(S)$ is the trivial group. The *minimal base size* for G, denoted $b(G)$, is the size of the smallest base for G.

If S is a base for G, then a permutation $\pi \in G$, viewed as a function $D \to D$, is uniquely determined by its restriction to S.

Existence of large quantum speedups for any sufficiently "small" permutation group

Let G be a permutation group on $[n]$ and f be a partial function. Then we can design a partial function g symmetric under G such that:

- $Q(g) \leq Q(f) + b(G)$.
- $R(g) \geq R(f)$.

20

Primitive groups

Block system
Let G be a permutation group on domain D. A partition $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ of D is called a *block system* for G if $\mathcal{B}$ is G-invariant. Formally, this means that $\pi(B_i) \in \mathcal{B}$ for every $\pi \in G$ and $i \in [k]$, where $\pi(B_i) := \{\pi(x) : x \in B_i\}$. The sets B_i are called *blocks* for G.

Example:
Consider the action of the group of linear transformations on the nonzero elements of a vector space. The set of lines through the origin (i.e., nonzero scalar multiples of a nonzero vector) form a block system.

Primitive group
Let G be a transitive permutation group on domain D. G is called primitive if the only block systems for G are $\{D\}$ and the partition of D into singletons. A permutation group that is not primitive is called imprimitive.

In other words, it preserves no non-trivial partitions of D.

Primitive groups

Existence of large quantum speedups for any sufficiently "small" permutation group

Let G be a permutation group on $[n]$ and f be a partial function. Then we can design a partial function g symmetric under G such that:

- $Q(g) \leq Q(f) + b(G)$.
- $R(g) \geq R(f)$.

Theorem (Liebeck 1984)

- For primitive groups either $b(G)$ is really small, or
- G contains the symmetries of a p-uniform hypergraph.

Existence of large quantum speedups for any sufficiently "small" permutation group

Let G be a permutation group on $[n]$ and f be a partial function. Then we can design a partial function g symmetric under G such that:

- $Q(g) \leq Q(f) + b(G)$.
- $R(g) \geq R(f)$.

Theorem (Liebeck 1984)

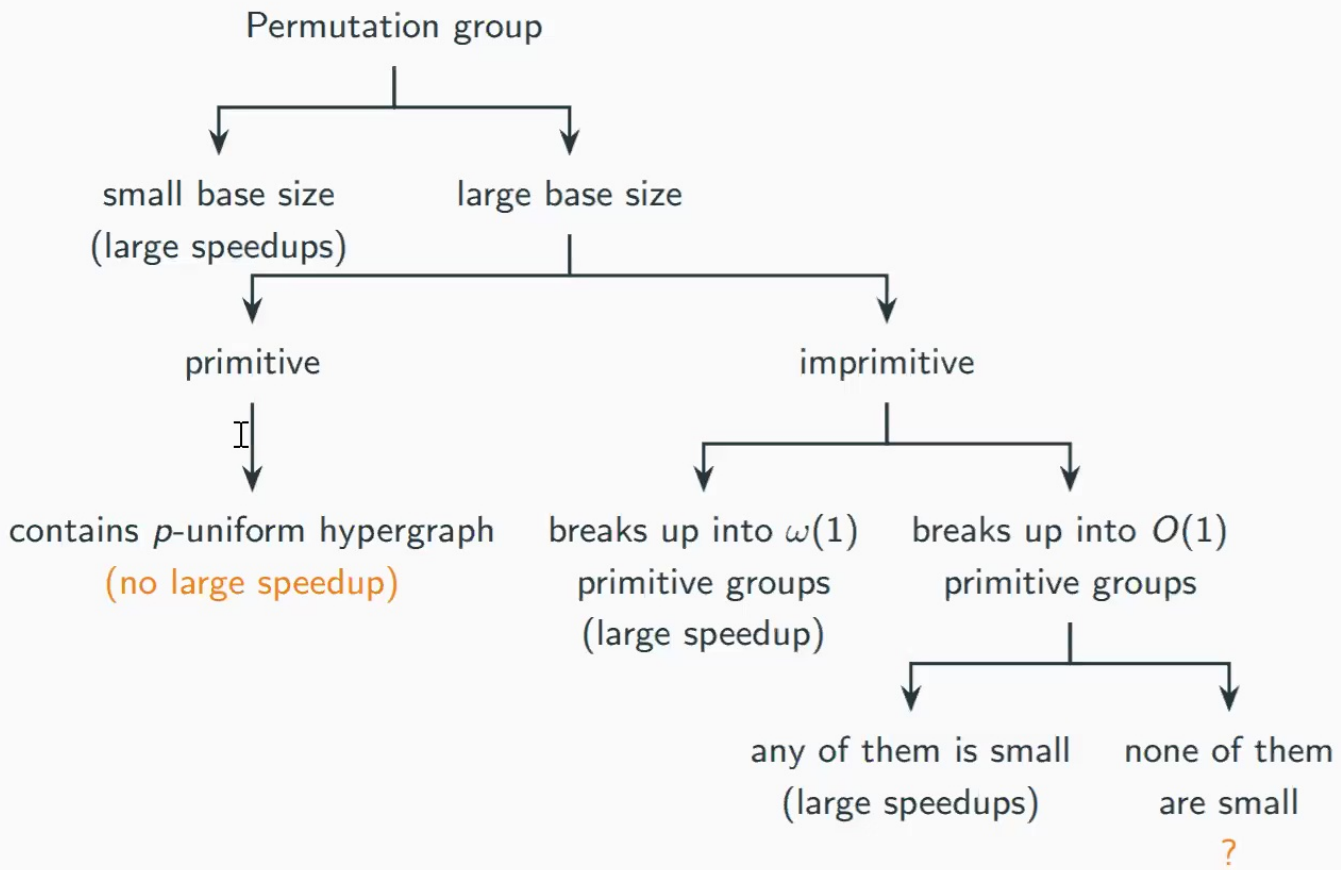
- For primitive groups either $b(G)$ is really small, or
- G contains the symmetries of a p-uniform hypergraph.

Full characterization of primitive permutation groups

Let G be a primitive permutation group. Then:

- If base size of G is $n^{o(1)}$, then there exists a class of partial functions f symmetric under G, $R(f) = Q(f)^{\omega(1)}$.
- If base size of G is $n^{\Omega(1)}$, then for all partial functions f symmetric under G, $R(f) = Q(f)^{O(1)}$.

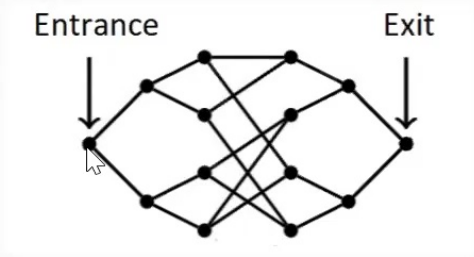
20



Here "large" denotes super-polynomial speedups.

Exponential speedups of graph property testing problem in the adjacency list model

Exponential speedups by welded trees

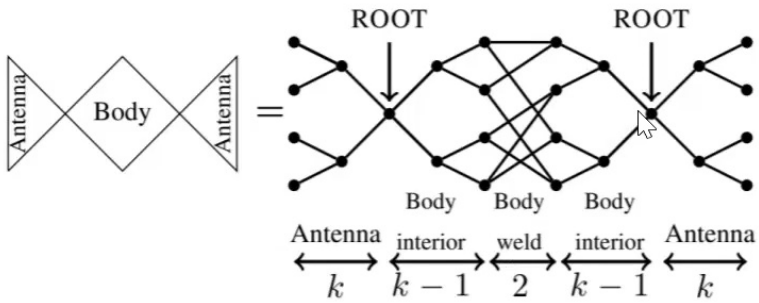


Welded trees

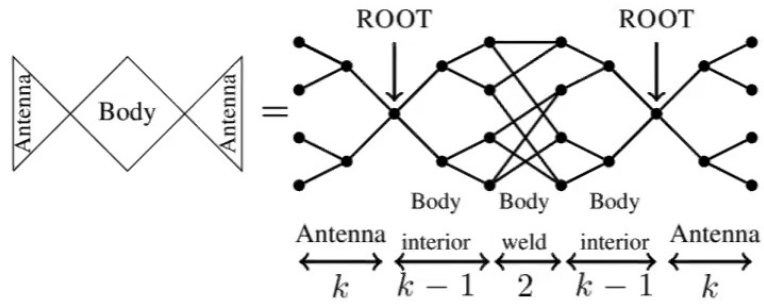
Welded trees problem
Given the adjacency list representation of two welded trees and given the entrance vertex, find the exit vertex.

Theorem: Childs, Cleve, Deotto, Farhi, Gutmann, Spielman 2003

Given a welded trees problem, a quantum walk algorithm finds the *exit* vertex in time $poly(n)$, but a randomized classical algo takes $2^{\Omega(n)}$ time.


Antenna
Body
Antenna
=
ROOT
ROOT
Body
Body
Body
Antenna
interior
weld
interior
Antenna
k
$k-1$
2
$k-1$
k

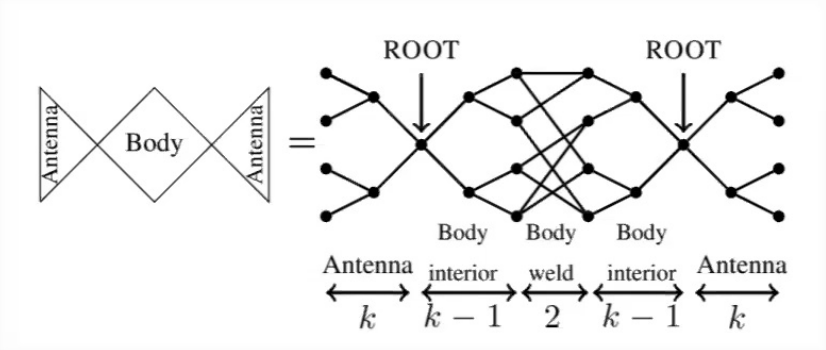
Candy graph



Candy graph

- Connect two large binary trees ("antenna") to the entrance and exit vertex.
- A random vertex is a leaf of the antenna with about 1/2 probability.

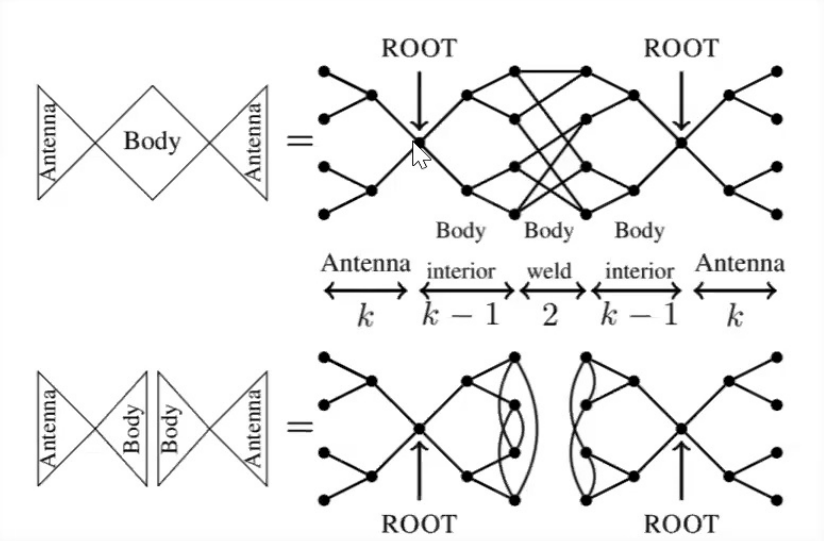
Graph property speedups



Candy graph

- Connect two large binary trees ("antenna") to the entrance and exit vertex.
- A random vertex is a leaf of the antenna with about 1/2 probability.
- From such a vertex it is easy to find the entrance.
- And then quantum algorithm can quickly find the exit vertex, whereas classical algorithm still needs exponential time to do so with high probability.

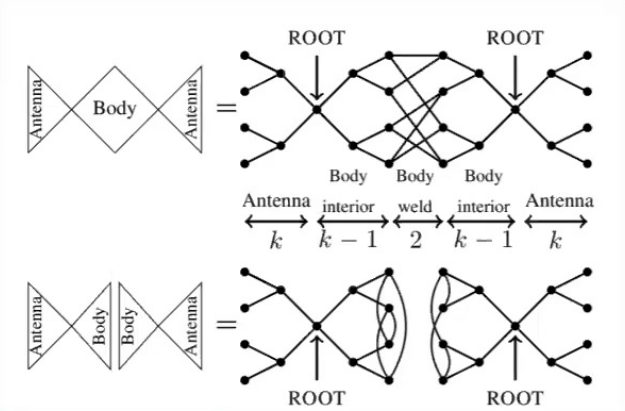
Turning into graph property testing

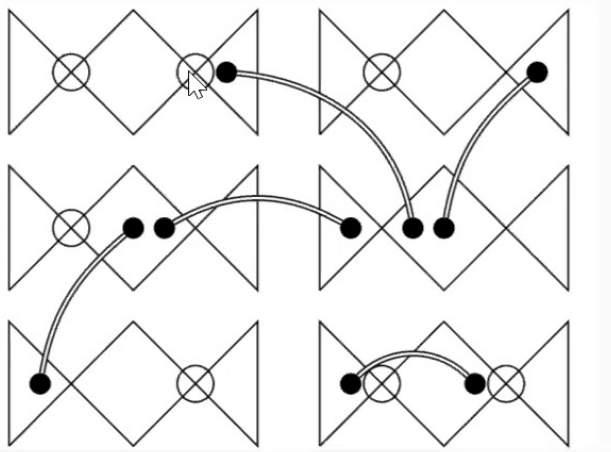


Candy graph

- Goal: Design a structure that is quantumly efficient, but classically hard to distinguish even from a ϵ-far graph.
- We have many copies of candy graphs.
- We allow self-loops and parallel edges.
- Almost half of the candy graphs have one root with self-loop. The other other half of the graphs have either no or two roots with self-loop.

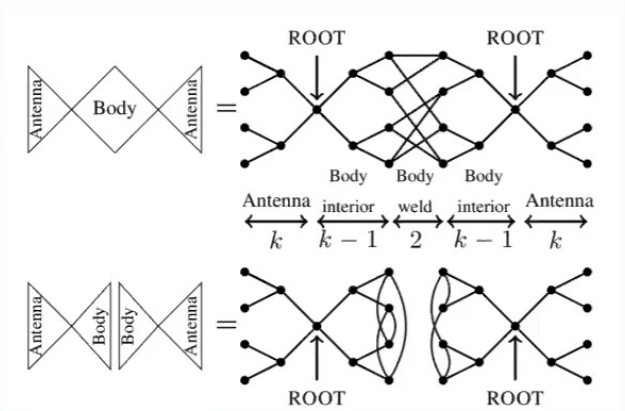
Graph property testing speedups

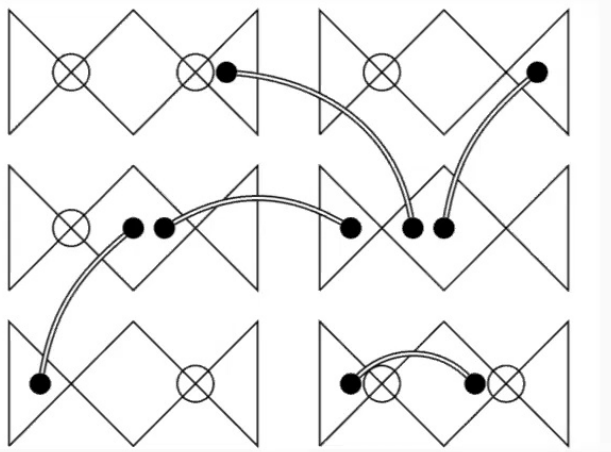




Property testing

Graph property testing speedups

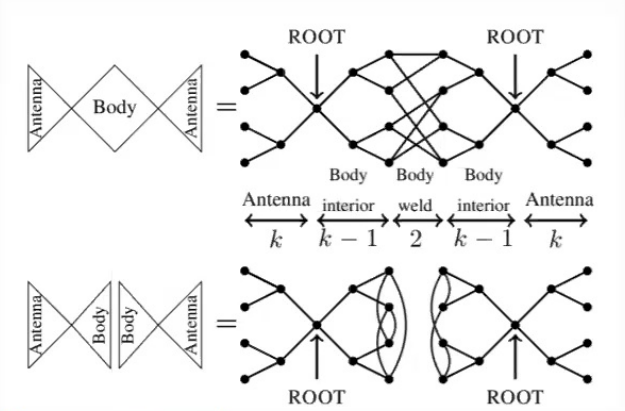


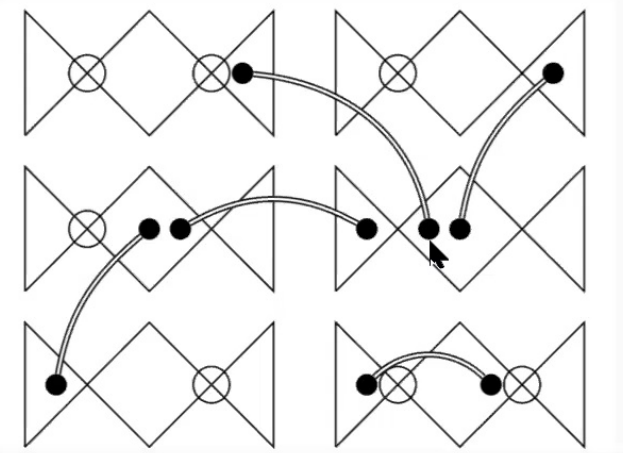


Property testing

- We have double edges from every non-root node to a node in the antenna graph satisfying certain property (works as advice).
- A graph has the property $\mathcal{P}_k$ if its single-edges form a graph that contains $2j(2^k - 1)$ disconnected instances of "candy" graphs for some $j \in \mathbb{N}$.

Graph property testing speedups



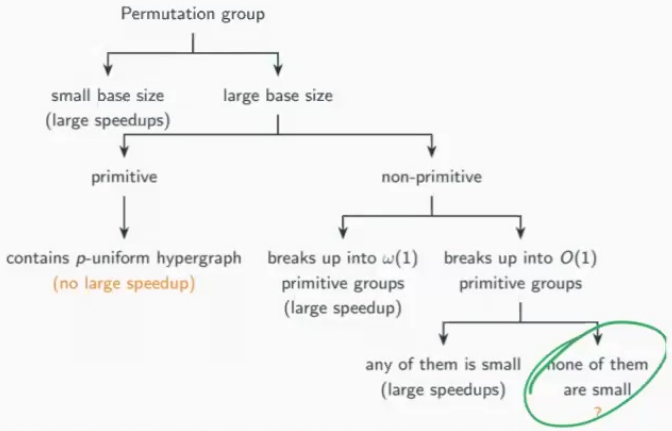


Property testing

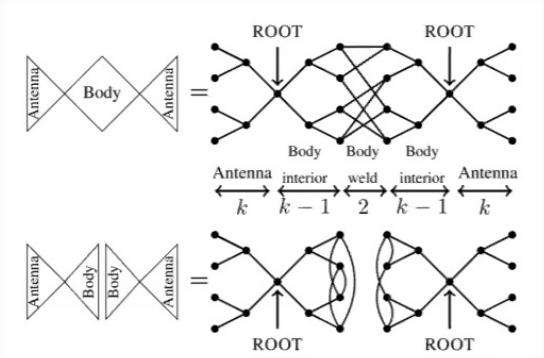
- We have double edges from every non-root node to a node in the antenna graph satisfying certain property (works as advice).
- A graph has the property $\mathcal{P}_k$ if its single-edges form a graph that contains $2j(2^k - 1)$ disconnected instances of "candy" graphs for some $j \in \mathbb{N}$.
- Given this "advice", the problem essentially reduces to testing a binary tree which can be done efficiently even classically.
- Yet, computing the advice require traversing the welded tree which is hard for any classical algorithm.

Open questions

Can we fully characterize super-polynomial speedups for all permutation groups?



Can we find exponential speed up for a practical graph problem in the adjacency list model?



More open questions...

- Super-polynomial speedups for monotone graph property i.e., those that remains satisfied when adding edges to a yes instance?
- What is the largest separation between $Q(f)$ and $R(f)$ for p-uniform hypergraph properties f? That is what is the largest k for which there exists such an f with $R(f) = \Omega(Q(f)^k)$?

Thank you so much for listening!!

Full version available at arXiv:2006:12760