Quantum Algorithms for Finding Constant-sized Sub-hypergraphs

François Le Gall*, Harumichi Nishimura† and Seiichiro Tani‡

*Graduate School of Information Science and Technology, the University of Tokyo.
†Graduate School of Information Science, Nagoya University.
‡NTT Communication Science Labs., NTT Corporation.

The 18th Conference on Quantum Information Processing

Graph Property Testing

Definition (Graph Property)

- Graph properties are those of graphs that are invariant under changing the labelings of vertices.
  (ex. connectedness, planarity)
- If a simple graph G is given as its adjacency matrix A_G, then whether G has a certain graph property or not can be expressed as a (transitive) Boolean function over $\binom{n}{2}$ elements in A_G.

Graph Property Testing

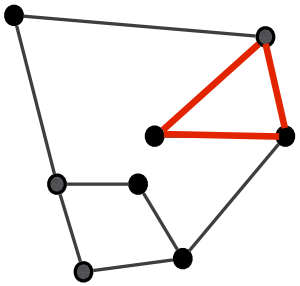
Decide if a graph $G = (V, E)$ has a graph property P with a minimum number of queries of the form "Is the pair (i,j) an edge of G?" ($=A_G[i, j]$))
(ignoring the cost of other operations.)

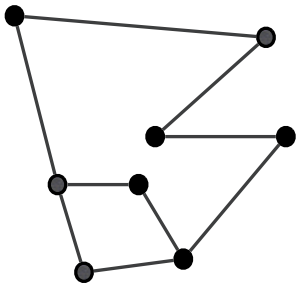
There are a long history of studies on this subject in classical computer science, particularly for monotone graph properties.

Triangle Finding

Triangle Finding Problem

Given a graph, decide with high probability (say, $\geq 2/3$) if it contains a triangle as a subgraph by making a minimum number of queries.





no triangle

This is an particularly important problem well studied since a fast triangle finding algorithm in the sense of time complexity would compute/solve fast

- Boolean matrix multiplication
- Max 2 -SAT

As a first step, query-efficient algorithms are worth studying.

Triangle Finding (Cont'd)

Classical Case $\Omega(n^2)$ queries — we need to query almost all.

Quantum Case $O(\sqrt{\binom{n}{3}}) = O(n^{1.5})$ can obtained simply by applying Grover's quantum search algorithm.

Moreover, a series of improvements have been made by introducing novel general-purpose quantum techniques.

The triangle finding is one of the central problems that have advanced quantum algorithm/complexity theory.

Quantum Algorithms for Triangle Finding

The trivial quantum upper bound is $O(n^{1.5})$ queries.

- $\tilde{O}(n^{1.3})$ queries [Magniez-Santha-Szegedy, SODA2005]
  by a new application of quantum walk.
- $O(n^{35/27})$ queries [Belovs, STOC2012] (35/27=1.296...)
  by introducing the learning graph technique.
- $O(n^{9/7})$ queries [Lee-Magniez-Santha, SODA2013] (9/7=1.285...)
  by improving the learning graph technique.
- $\tilde{O}(n^{9/7})$ queries (simpler algorithm) [Jeffery-Kothari-Magniez, SODA2013]
  by introducing the concept of nested quantum walk.
- $O(n^{5/4})$ queries [LeGall, FOCS2014] (5/4=1.25)
  by combinatorial arguments + quantum walk.

These works have developed new quantum techniques for general purposes.
Along this line of research,
we consider a generalization of triangle finding to the hypergraph case.

Hypergraphs

Definition (3-uniform Hypergraphs)

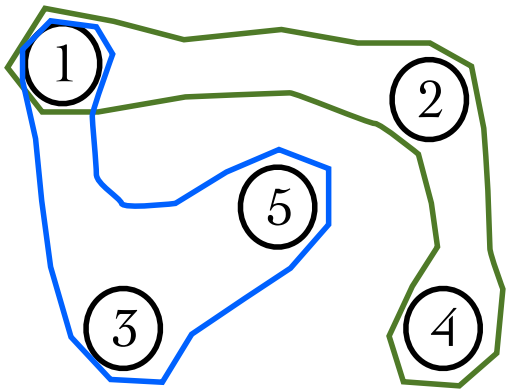
An undirected 3-uniform hypergraph is a pair (V, E), where

- V is a finite set (the set of vertices),
- $E \subseteq V \times V \times V$ is the set of hyperedges, i.e., unordered triples of elements in V.

Example

$V = \{1, 2, 3, 4, 5\}$

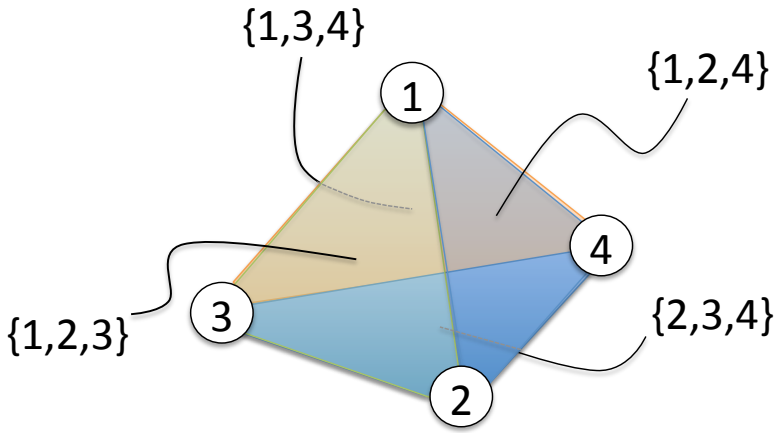
$E = \{\{1, 2, 4\}, \{1, 3, 5\}\}$



Note that we can define k-uniform hypergraphs, but we only deal with 3-uniform case in this talk.

4-Clique over a 3-Uniform Hypergraph

4-clique is a complete 3-uniform hypergraph on 4 vertices:
(a generalization of a triangle)



Example

Ex. $\{1,2,3\}, \{1,2,4\}, \{1,3,4\}, \{2,3,4\}$ are all hyperedges.

4-Clique Finding Problem

Given a hypergraph G, decide with high probability
if it contains a 4-clique as a subhypergraph by making a minimum number of queries of the form: "Is the triple $\{i,j,k\}$ an hyperedge of G?"

This problem is closely related to Max-3SAT or multiplication of tensors.

Our Results: Finding 4-Clique in a 3-uniform Hypergraph

Theorem (4-clique Finding Quantum Algorithm)

There exists a quantum algorithm that detects with high probability if the input 3-uniform hypergraph on n vertices has a 4-clique as a subhypergraph (and finds a 4-clique if it exists), by making $\tilde{O}(n^{241/128}) = O(n^{1.883})$ queries.

- Better than naïve Grover search over the $\binom{n}{4}$ combinations of vertices, which only gives $O(n^2)$ queries.
- Our algorithm actually works for finding any constant-sized subhypergraph
  (the quantum query complexity depends on the subhypergraph).

Our Results: Technical Outline

1. Extend the idea of the triangle finding algorithm by [Lee-Magniez-Santha, SODA05] to the hypergraph case.
   But the analysis gets too complicated to be done.
2. Then cast the extended idea to the framework of nested quantum walk introduced by [Jeffery-Kothari-Magniez, SODA05].
   Still, need to somehow handle undesirable cases that is unique in the hypergraph case.
3. Finally heavily use the concentration theorem over hypergeometric distribution to show that the undesirable cases rarely happen.

Applications: Ternary Associativity Testing

Let X be a finite set with $|X| = n$. A ternary operator $\mathcal{F}$ from $X \times X \times X$ to X is said to be *associative* if
$\mathcal{F}(\mathcal{F}(a,b,c),d,e) = \mathcal{F}(a,\mathcal{F}(b,c,d),e) = \mathcal{F}(a,b,\mathcal{F}(c,d,e))$
holds for every 5-tuple $(a,b,c,d,e) \in X^5$.

Theorem (Ternary Associativity Testing)

There exists a quantum algorithm that determines if $\mathcal{F}$ is associative with high probability using $\tilde{O}(n^{169/80}) = \tilde{O}(n^{2.1125})$ queries.

Proof.

First transform ternary associativity testing into the problem of finding a certain subhypegraph of constant size. Then, we apply our algorithm. □

Lee-Magniez-Santha Algorithm for Finding a Triangle

Let us assume $\{a_1, a_2, a_3\}$ forms a triangle (on the given ordinary graph). The algorithm samples objects recursively.

- Sample a set $V_1 \subseteq V$ with size v_1 of candidates for a_1.
- To check if V_1 is marked,
  Sample a set $V_2 \subseteq V$ with size v_2 of candidates for a_2.
- To check if V_2 is marked,
  sample a set $V_3 \subseteq V$ with size v_3 of candidates for a_3.
- To check if V_3 is marked,
  sample a set $E_{12} \subseteq V_1 \times V_2$ with size e_{12} of candidates for $\{a_1, a_2\}$.
- To check if E_{12} is marked,
  sample a set $E_{13} \subseteq V_1 \times V_3$ with size e_{13} of candidates for $\{a_1, a_3\}$.
- To check if E_{13} is marked,
  sample a set $E_{23} \subseteq V_2 \times V_3$with size e_{23} of candidates for $\{a_2, a_3\}$.
- Check if a triangle $\{a_1, a_2, a_3\}$ in $E_{12} \cup E_{13} \cup E_{23}$.

Our First Trial for Fnding 4-clique

Let $\{a_1, a_2, a_3, a_4\}$ be a 4-clique.

- Sample a set $V_1 \subseteq V$ with size v_1 of candidates for a_1 from V.
- To check if V_1 is marked,
  sample a set $V_2 \subseteq V$ with size v_2 of candidates for a_2 from V,
- ...
- To check if V_4 is marked,
  sample a set $E_{123} \subseteq V_1 \times V_2 \times V_3$ with size e_{123} of candidates for $\{a_1, a_2, a_3\}$.
- To check if E_{123} is marked,
  sample a set $E_{124} \subseteq V_1 \times V_2 \times V_4$ with size e_{124} of candidates for $\{a_2, a_3, a_4\}$.
- ...
- Check if a 4-clique $\{a_1, a_2, a_3, a_4\}$ is in $E_{123} \cup E_{124} \cup E_{134} \cup E_{234}$.

where $v_1, v_2, v_3, v_4, e_{123}, e_{124}, e_{134}, e_{234}$ are parameters to be optimized.

But this gives no improvement over the trivial bound $\sqrt{\binom{n}{4}} = O(n^2)$ via Grover's algorithm.

Our strategy for finding 4-clique (1/3)

Let $\{a_1, a_2, a_3, a_4\}$ be a 4-clique. Sampling is actually recursive.

1. Sample a set $V_1 \subseteq V$ with size v_1 of candidates for a_1.
2. To check if V_1 is marked,
   sample a set $V_2 \subseteq V$ with size v_2 of candidates for a_2.
3. To check if V_2 is marked,
   sample a set $V_3 \subseteq V$ with size v_3 of candidates for a_3.
4. To check if V_3 is marked,
   sample a set $V_4 \subseteq V$ with size v_4 of candidates for a_4.

   ...

   ...

   ...

Our strategy for finding 4-clique (2/3)

5 To check if V_4 is marked,
sample a set of $F_{12} \subseteq V_1 \times V_2$ with size f_{12} of candidates for $\{a_1, a_2\}$.

6 To check if F_{12} is marked,
sample a set of $F_{13} \subseteq V_1 \times V_3$ with size f_{13} of candidates for $\{a_1, a_3\}$.

7 To check if F_{14} is marked,
sample a set of $F_{14} \subseteq V_1 \times V_4$ with size f_{14} of candidates for $\{a_1, a_4\}$.

...

10 To check if F_{24} is marked,
sample a set of $F_{34} \subseteq V_3 \times V_4$ of candidates for $\{a_3, a_4\}$.

Our strategy for finding 4-clique (3/3)

11 To check if F_{34} is marked,
sample a set of E_{123} with size e_{123} of candidates for $\{v_1, v_2, v_3\}$
by picking a pair from each of F_{12}, F_{23}, F_{13} so that they form a triple.

....

14 To check if E_{134} is marked,
sample a set of E_{234} with size e_{234} of candidates for $\{v_2, v_3, v_4\}$
by picking a pair from each of F_{23}, F_{24}, F_{34} so that they form a triple.

15 Check if $E_{123} \cup E_{124} \cup E_{134} \cup E_{234}$ contains a 4-clique.

Outline of Complexity Analysis

This strategy can be cast as recursive quantum-walk-based search. Then, the query complexity is

$$\tilde{O}\left(\mathsf{S} + \sum_{t=1}^{m}\left(\prod_{r=1}^{t}\frac{1}{\sqrt{\varepsilon_r}}\right)\frac{1}{\sqrt{\delta_t}}\mathsf{U}_t\right),$$

where S, U_t, δ_t and ε_r are evaluated as follows:

- $\mathsf{S} = \sum_{\{i,j,k\}\in\Sigma_3} e_{ijk}$;
- for $t \in \{1,\ldots,m\}$, (i) if $s_t = \{i\}$, then $\delta_t = \Omega(\frac{1}{v_i})$, $\varepsilon_t = \Omega(\frac{v_i}{n})$ and $\mathsf{U}_t = \tilde{O}\left(1 + \sum_{\{j,k\}:\{i,j,k\}\in\Sigma_3}\frac{e_{ijk}}{v_i}\right)$; (ii) if $s_t = \{i,j\}$, then $\delta_t = \Omega(\frac{1}{f_{ij}})$, $\varepsilon_t = \Omega(\frac{f_{ij}}{v_i v_j})$ and $\mathsf{U}_t = \tilde{O}\left(1 + \sum_{k:\{i,j,k\}\in\Sigma_3}\frac{e_{ijk}}{f_{ij}}\right)$; (iii) if $s_t = \{i,j,k\}$, then $\delta_t = \Omega(\frac{1}{e_{ijk}})$, $\varepsilon_t = \Omega(\frac{e_{ijk}v_i v_j v_k}{f_{ij}f_{ik}f_{jk}})$ and $\mathsf{U}_t = O(1)$.

Optimizing parameters v_i, f_{ij}, e_{ijk} gives $\tilde{O}(n^{241/128}) = O(n^{1.883})$ queries.

Crux of Analysis

Difficulties

Steps 11-14 randomly choose e_{ijk} triples from the set

$$\Gamma_{ijk} = \{(u, v, w) \mid (u, v) \in F_{ij}, (u, w) \in F_{ik} \text{ and } (v, w) \in F_{jk}\}.$$

The difficulties here are:
The size and structure of Γ_{ijk} vary depending on the sets F_{ij}, F_{jk}, F_{ik}, and thus they may be significantly changed by updating F_{ij} many times.

Our solution

We proved that F_{ij} is almost uniformly distributed by using concentration theorems for hypergeometric distributions. This implies that,
with exponentially small error, the size of Γ_{ijk} lies around its average and its structure is very regular,
which effectively makes it possible to analyze the complexity on average.

Conclusion

- We considered a generalization of Triangle Finding problem to the 3-uniform hypergraphs.
- For finding a 4-clique, we obtained a quantum algorithm with query complexity $O(n^{1.883})$, beating the $O(n^2)$-query trivial quantum algorithm.
- More generally, we developed a framework that give an efficient quantum algorithms for finding any constant-sized subhypergraph.
- For this, we designed a general technique for handling nested quantum walk over graphs of non-fixed size.

Open Problems

- Further improvements of our complexity?
- Can generalize our techniques to d-uniform hyper graphs ($d \geq 3$)?
- Other applications of our techniques?

Query Complexity Model (a.k.a. oracle model)

Definition (Classical Case)

- An input hypergraph $G = (V, E)$ is given as an oracle.

  Our case

  Oracle $= \left\{ h_{ijk} \in \{T, F\} \colon i < j < k, (i, j, k) \in V \times V \times V \right\}$.

- Algorithms need to make queries to the oracle to get input.

  Our case

  For the query $(\{i, j, k\}, ?)$, we receive the answer $(\{i, j, k\}, h_{ijk})$.

  Algorithm $\xrightarrow{(\{i,j,k\},?)}$ Oracle $\xrightarrow{(\{i,j,k\},T)}$ Algorithm

- Minimize # of queries, ignoring the cost of other operations.

  Our case

  The number of required queries is trivially at most $\binom{n}{3} = O(n^3)$.

Query Complexity Model (a.k.a. oracle model)

Definition (Quantum Case)

- An input hypergraph $G = (V, E)$ is given as an oracle.

  Our case

  Oracle $= \left\{ h_{ijk} \in \{T, F\} \colon i < j < k, (i, j, k) \in V \times V \times V \right\}$.

- Algorithms need to make quantum queries to the oracle to get input.

  Our case

  - Quantum queries are superpositions of many classical queries, and the answers are those of the corresp. classical answers: a query $\sum \alpha_{i,j,k} |\{i, j, k\}, ?\rangle$, and the answer $\sum \alpha_{i,j,k} \left|\{i, j, k\}, h_{ijk}\right\rangle$.
  - Note: a classical query can be simulated by a quantum query: Set $\alpha_{ijk} = 1$ and $\alpha_{pqr} = 0$ for all $(p, q, r) \neq (i, j, k)$.

- Minimize # of quantum queries, ignoring the cost of other operations.

Search with Random Walk

Search Problem

Given a Boolean function f over the domain X onto $\{0,1\}$,
find a solution $x \in X$ such that $f(x) = 1$.

Simple Sampling Idea

- Sample a subset $Y_1 \subseteq X$ of size r.
- Check if Y_1 contains a solution; if it indeed does, we are done.
- Otherwise, we update Y_1 to Y_2 by replacing a random element in Y with a new element that is chosen at random from $X \setminus Y_1$.
  (Y_1 and Y_2 differ only by one element)
- Check if Y_2 contains a solution; if it indeed does, we are done.
- Otherwise, we update Y_2 to Y_3 by replacing...

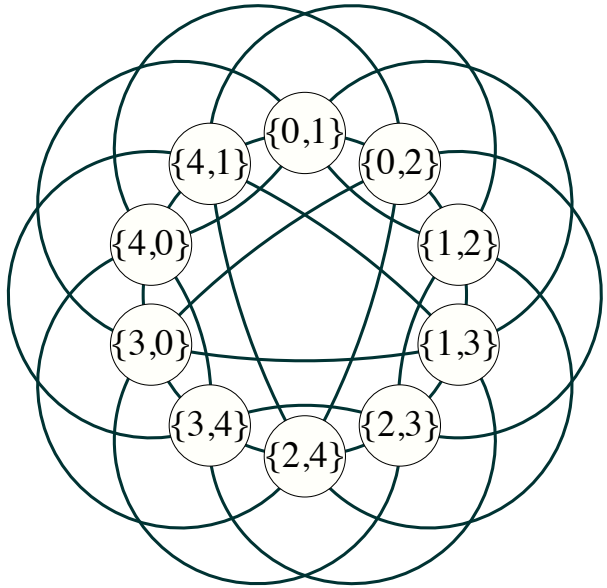
We can regard the sequence $Y_1 \to Y_2 \to Y_3 \to \cdots$ as random walks over the graph whose nodes are subsets of size r of X.

Johnson Graph

Definition (Johnson graph $J(n, r) = (V, E)$)

- V is the collection of all r-sized subsets of $[n]$, so that $|V| = \binom{n}{r}$. (Corresponding to sampling r-sized subsets from X with $|X| = n$).
- For every vertex pairs $U, T \in V$, the pair $\{U, V\}$ is an edge (an element in E) if and only if U and T differ only by one element.

ex.) $J(5, 2)$ looks like $\longrightarrow$.



Fact.

The spectral gap of $J(n, r)$ is $\Theta(1/r)$.

The spectral gap of the graph affects the hitting time of random walk over $J(n, r)$.

Search with Random Walk

Let us say that the nodes containing a solution is marked.

Fact.

If the underlying graph has spectral gap δ and the fraction of marked nodes is ϵ, then the hitting time (the number of steps required to find a marked node with high probability) is $O(\frac{1}{\delta \cdot \epsilon})$.

Corollary

The total cost for finding a solution is

$$S + \frac{1}{\epsilon}\left(\frac{1}{\delta}U + C\right),$$

S: cost of initial sampling (initial queries)
U: cost of one step random walk (addition queries)
C: cost of checking if the node is marked. (additional queries).
(Here we perform checking procedure every $1/\delta$ steps.)

Search with Quantum Walk

[Ambanis, Szegedy, Magniez-Nayak-Roland-Santha]

Let us say that the nodes containing a solution is marked.

Fact.

If the underlying graph has spectral gap δ and the fraction of marked nodes is ϵ, then the number of steps required to find a marked node is ~~$O(\frac{1}{\delta\cdot\epsilon})$~~ $O(\sqrt{\frac{1}{\delta\cdot\epsilon}})$ with high probability. Note $\frac{1}{\delta\cdot\epsilon} \geq \sqrt{\frac{1}{\delta\cdot\epsilon}}$.

This implies that the total cost for finding a solution is

$$\cancel{S + \frac{1}{\epsilon}\left(\frac{1}{\delta}U + C\right)} \quad S + \frac{1}{\sqrt{\epsilon}}\left(\frac{1}{\sqrt{\delta}}U + C\right),$$

where

S: cost of initial sampling (initial queries)

U: cost of one step random walk (addition queries)

C: cost of checking if the node is marked. (additional queries).

(Here we perform checking procedure every $1/\delta$ steps.)