The Average-Case Complexity of Counting Cliques in Erdős-Rényi Hypergraphs

Enric Boix-Adserà, Matthew Brennan and Guy Bresler

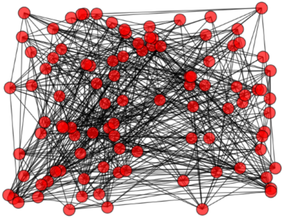
MIT

June 29, 2020

Setup

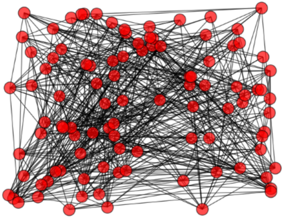
Setup

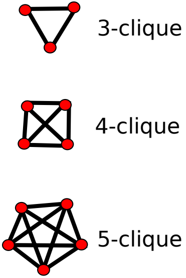
- $\mathcal{G}_s(n, p)$ is the Erdős-Rényi hypergraph:
  1. A distribution over random n-vertex s-uniform hypergraphs
  2. Each s-subset of vertices is a hyperedge independently with prob. p



Setup

- $\mathcal{G}_s(n, p)$ is the Erdős-Rényi hypergraph:
  1. A distribution over random n-vertex s-uniform hypergraphs
  2. Each s-subset of vertices is a hyperedge independently with prob. p

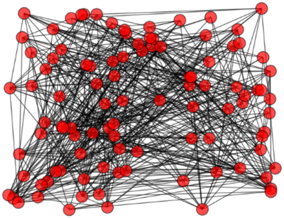


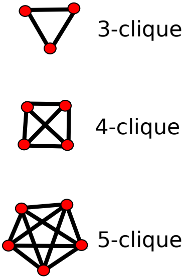


- Counting k-cliques in G is the problem of outputting the exact number of complete k-vertex subgraphs in G w.h.p.

Setup

- $\mathcal{G}_s(n, p)$ is the Erdős-Rényi hypergraph:
  1. A distribution over random n-vertex s-uniform hypergraphs
  2. Each s-subset of vertices is a hyperedge independently with prob. p





- Counting k-cliques in G is the problem of outputting the exact number of complete k-vertex subgraphs in G w.h.p.
- Constant clique size $k = \Theta(1)$ throughout

Our Question: How does the optimal running time T for counting k-cliques in $\mathcal{G}_s(n, p)$ trade off with n, p and s?

Clique Problems on Erdős-Rényi Graphs

Clique Problems on Erdős-Rényi Graphs

Algorithmic barriers in clique problems on $\mathcal{G}(n, p)$ studied for decades

Clique Problems on Erdős-Rényi Graphs

Algorithmic barriers in clique problems on $\mathcal{G}(n, p)$ studied for decades

Planted Clique: Find a k-clique planted in $\mathcal{G}(n, 1/2)$

- Lower bounds for greedy, SOS hierarchy, SQ algorithms, resolution (Jerrum '92, Barak et al. '16, Feldman et al. '13, Atserias et al. '18, etc.)
- Hardness implies stat-comp gaps (Berthet-Rigollet '13, Koiran-Zouzias '14, Hajek-Wu-Xu '15, Ma-Wu '15, B.-Bresler-Huleihel '18, '19, etc.)

Clique Problems on Erdős-Rényi Graphs

Algorithmic barriers in clique problems on $\mathcal{G}(n, p)$ studied for decades

Planted Clique: Find a k-clique planted in $\mathcal{G}(n, 1/2)$

Find Large Cliques: Find largest clique possible in $\mathcal{G}(n, 1/2)$

- Lower bounds for Metropolis, greedy
  (Karp '76, Grimmet-Mcdiarmid '75, Mcdiarmid '84, Jerrum '92, etc...)

Clique Problems on Erdős-Rényi Graphs

Algorithmic barriers in clique problems on $\mathcal{G}(n, p)$ studied for decades

Planted Clique: Find a k-clique planted in $\mathcal{G}(n, 1/2)$

Find Large Cliques: Find largest clique possible in $\mathcal{G}(n, 1/2)$

Find Critical Cliques: Find a k-clique in $\mathcal{G}(n, n^{-\alpha})$

- Lower bounds for AC_0 and monotone circuits
  (Rossman '08, Rossman '10)

Clique Problems on Erdős-Rényi Graphs

Algorithmic barriers in clique problems on $\mathcal{G}(n, p)$ studied for decades

Planted Clique: Find a k-clique planted in $\mathcal{G}(n, 1/2)$

Find Large Cliques: Find largest clique possible in $\mathcal{G}(n, 1/2)$

Find Critical Cliques: Find a k-clique in $\mathcal{G}(n, n^{-\alpha})$

Many Others: e.g. Gamarnik-Sudan '14, Coja-Oghlan-Efthymiou '15, Rahman-Virag '17

Why k-Clique Counting?

Ideally would base average-case hardness on worst-case hardness e.g. prove planted clique is NP-hard

Why k-Clique Counting?

Ideally would base average-case hardness on worst-case hardness e.g. prove planted clique is NP-hard, **BUT**

- Barriers against worst-case to average-case reductions for NP-complete problems (Feigenbaum-Fortnow '93, Bogdanov-Trevisan '05)

Why k-Clique Counting?

Ideally would base average-case hardness on worst-case hardness e.g. prove planted clique is NP-hard, **BUT**

- Barriers against worst-case to average-case reductions for NP-complete problems (Feigenbaum-Fortnow '93, Bogdanov-Trevisan '05)

Work-around: Counting k-cliques **is in P** – and we show (fine-grained) average-case hardness from worst-case hardness assumption

Plan for Rest of the Talk

1. *Overview of algorithmic results:* Previously-known worst-case algorithms and our algorithms on $\mathcal{G}_s(n, p)$
2. *Main hardness result:* Partial answer to our question
3. *Proof sketch:* Worst-case to average-case reduction
4. *Open Problems:* Error tolerance, approximation hardness and more

Algorithms for Counting k-Cliques

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	Same as worst-case?	Same as worst-case?

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n,p)$ $p = \Theta(1)$	Same as worst-case?	Same as worst-case?
Sparse $\mathcal{G}_s(n,p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours)	Fast matrix mult. speedup (ours)

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega\lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	Same as worst-case?	Same as worst-case?
Sparse $\mathcal{G}_s(n, p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours)	Fast matrix mult. speedup (ours)

1. **Faster algorithms for sparse Erdos-Renyi than worst-case!**

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	Same as worst-case?	Same as worst-case?
Sparse $\mathcal{G}_s(n, p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours)	Fast matrix mult. speedup (ours)

1. **Faster algorithms for sparse Erdos-Renyi than worst-case!**
2. **What can be improved?** Under ETH, worst-case runtime $n^{\Omega(k)}$

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	Same as worst-case?	Same as worst-case?
Sparse $\mathcal{G}_s(n, p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours)	Fast matrix mult. speedup (ours)

1. **Faster algorithms for sparse Erdos-Renyi than worst-case!**
2. **What can be improved?** Under ETH, worst-case runtime $n^{\Omega(k)}$
3. **How about average-case?** Our main result: lower-bounds on run-time based on worst-case hardness

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	**Same as worst-case!**	**Same as worst-case!**
Sparse $\mathcal{G}_s(n, p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours)	Fast matrix mult. speedup (ours)

1. **Faster algorithms for sparse Erdos-Renyi than worst-case!**
2. **What can be improved?** Under ETH, worst-case runtime $n^{\Omega(k)}$
3. **How about average-case?** Our main result: lower-bounds on run-time based on worst-case hardness

Algorithm run-times

	Hypergraphs ($s \geq 3$)	Graphs ($s = 2$)
Worst-case	$O(n^k)$ (exhaustive search)	$O(n^{\omega \lfloor k/3 \rfloor})$ (Nesetril-Poljak '85)
Dense $\mathcal{G}_s(n, p)$ $p = \Theta(1)$	**Same as worst-case!**	**Same as worst-case!**
Sparse $\mathcal{G}_s(n, p)$ $p = \Theta(n^{-\alpha})$	$O(n^{\tau+1-\alpha\binom{\tau+1}{s}})$ τ largest s.t. $\alpha\binom{\tau}{s-1} < 1$ (ours) **Optimal in some regimes!**	Fast matrix mult. speedup (ours)

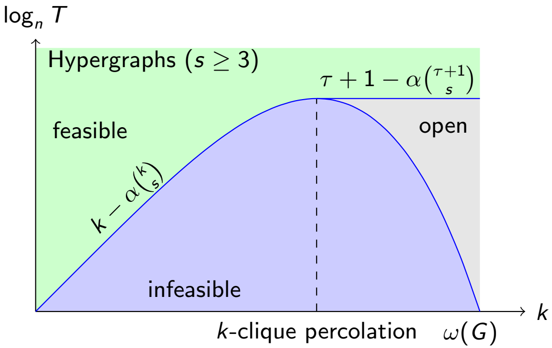
1. **Faster algorithms for sparse Erdos-Renyi than worst-case!**
2. **What can be improved?** Under ETH, worst-case runtime $n^{\Omega(k)}$
3. **How about average-case?** Our main result: lower-bounds on run-time based on worst-case hardness

Main Result: Average-case lower bounds from worst-case assumptions

Assumption: $O(n^k)$ for $s \geq 3$ and $O(n^{\omega \lfloor k/3 \rfloor})$ for $s \geq 2$ are the optimal **worst-case** running times

Results for Hypergraphs ($s \geq 3$)

Feasible pairs of clique sizes k and runtimes T at density $p = \Theta(n^{-\alpha})$

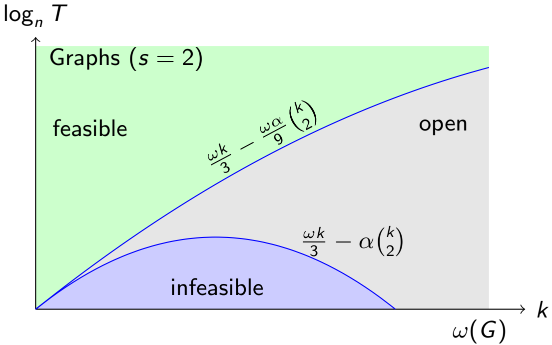


Infeasible assuming worst-case $O(n^k)$-time algorithm is optimal

Match up to k-clique percolation threshold!

Results for Graphs ($s = 2$)

Feasible pairs of clique sizes k and runtimes T at density $p = \Theta(n^{-\alpha})$



Infeasible assuming worst-case $O(n^{\omega k/3})$-time algorithm is optimal

Optimal exponent is of the form $\frac{\omega k}{3} - C\binom{k}{2}$ for $\frac{\omega\alpha}{9} \leq C \leq \alpha$

Main Theorem: Worst-Case to Average-Case Reduction

Given an alg A, let $T(A, n)$ denote its runtime on size-n inputs

Theorem

There is a slowdown factor

$$\Upsilon \asymp \left(p^{-1}(1-p)^{-1} \log n \log \log n\right)^{\binom{k}{s}}$$

s.t. for any alg A for k-clique counting with error probability less than $1/\Upsilon$ on hypergraphs drawn from $\mathcal{G}_s(n, p)$, there is an alg B that has error probability less than $1/3$ on any worst-case hypergraph s.t.

$$T(B, n) \leq (\log n) \cdot \Upsilon \cdot \left(T(A, nk) + (nk)^s\right)$$

Punchline: Intricate average-case complexity on $\mathcal{G}_s(n, p)$ follows from simple worst-case complexity!

Two-slide! proof sketch

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S| = k}} \prod_{i,j \in S} A_{ij}$$

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S|=k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S|=k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Ball et al. '17: Application to fine-grained complexity

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S| = k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Ball et al. '17: Application to fine-grained complexity

Goldreich-Rothblum '18: Application to k-clique counting

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S|=k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Ball et al. '17: Application to fine-grained complexity

Goldreich-Rothblum '18: Application to k-clique counting
Issue: want average-case distribution over $\{0,1\}^N$, *not over* $\mathbb{F}_q^N$

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S| = k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Ball et al. '17: Application to fine-grained complexity

Goldreich-Rothblum '18: Application to k-clique counting
Issue: want average-case distribution over $\{0,1\}^N$, *not over* $\mathbb{F}_q^N$

- Replace each $\mathbb{F}_q$-weighted edge with a **gadget** of unweighted edges

Background

k-clique count in graphs is a low-degree polynomial in adjacency matrix:

$$P(A) = \sum_{\substack{S \subset [n] \\ |S|=k}} \prod_{i,j \in S} A_{ij}$$

Lipton '89: Classic trick for worst- to avg-case reductions for polynomials

- Given low-degree polynomial $P : \mathbb{F}_q^n \to \mathbb{F}_q$, evaluating P on worst-case inputs reduces to evaluating it on average-case inputs. **Works only if finite field $\mathbb{F}_q$ is large enough.**

Ball et al. '17: Application to fine-grained complexity

Goldreich-Rothblum '18: Application to k-clique counting
Issue: want average-case distribution over $\{0, 1\}^N$, *not over* $\mathbb{F}_q^N$

- Replace each $\mathbb{F}_q$-weighted edge with a **gadget** of unweighted edges
- They get very good error tolerance, but artificial graph distribution. Does not seem possible to arrive at $\mathcal{G}_s(n, p)$ with their method

Proof ingredients

Main technical obstacle is to map random element of $\mathbb{F}_q^N$ to the $\mathcal{G}_s(n, p)$ distribution. Ingredients of our proof include:

Proof ingredients

Main technical obstacle is to map random element of $\mathbb{F}_q^N$ to the $\mathcal{G}_s(n, p)$ distribution. Ingredients of our proof include:

1. Reduction to k-partite $\mathcal{G}_s(n, p)$ (using inclusion-exclusion principle)

Proof ingredients

Main technical obstacle is to map random element of $\mathbb{F}_q^N$ to the $\mathcal{G}_s(n, p)$ distribution. Ingredients of our proof include:

1. Reduction to k-partite $\mathcal{G}_s(n, p)$ (using inclusion-exclusion principle)
2. Using special structure of k-partite k-clique counting polynomial (color-coding trick)

Proof ingredients

Main technical obstacle is to map random element of $\mathbb{F}_q^N$ to the $\mathcal{G}_s(n, p)$ distribution. Ingredients of our proof include:

1. Reduction to k-partite $\mathcal{G}_s(n, p)$ (using inclusion-exclusion principle)
2. Using special structure of k-partite k-clique counting polynomial (color-coding trick)
3. Tight analysis of convergence of biased binary expansions modulo a prime (using Fourier analysis)

Proof ingredients

Main technical obstacle is to map random element of $\mathbb{F}_q^N$ to the $\mathcal{G}_s(n, p)$ distribution. Ingredients of our proof include:

1. Reduction to k-partite $\mathcal{G}_s(n, p)$ (using inclusion-exclusion principle)
2. Using special structure of k-partite k-clique counting polynomial (color-coding trick)
3. Tight analysis of convergence of biased binary expansions modulo a prime (using Fourier analysis)
4. Keeping field size q small (using Chinese remaindering)

Summary of contributions & open problems

Contributions

- Studied k-clique counting on Erdős-Rényi hypergraphs
- Faster algorithms in sparse regime
- Average-case hardness based on worst-case hardness
  - Differs from other hardness results for clique problems on Erdos-Renyi graphs!
  - Tight in dense regime
  - Tight in some parts of sparse regime

Summary of contributions & open problems

Contributions

- Studied k-clique counting on Erdős-Rényi hypergraphs
- Faster algorithms in sparse regime
- Average-case hardness based on worst-case hardness
  - Differs from other hardness results for clique problems on Erdos-Renyi graphs!
  - Tight in dense regime
  - Tight in some parts of sparse regime

Some open problems

- Some regimes where upper bounds don't match lower bounds
- Hardness for approximating number of k-cliques?
- Improving error tolerance of reduction?