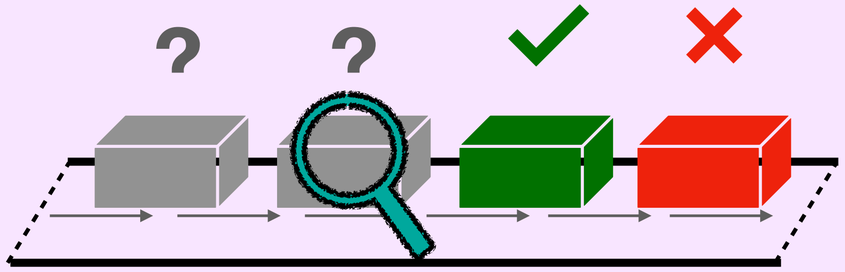
Quality Control in Sublinear Time

Cassandra Marcussen

Based on joint work with Ronitt Rubinfeld and Madhu Sudan



Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

Motivating Question

Given specific instance x & average-case algorithm $\mathscr{A}$:

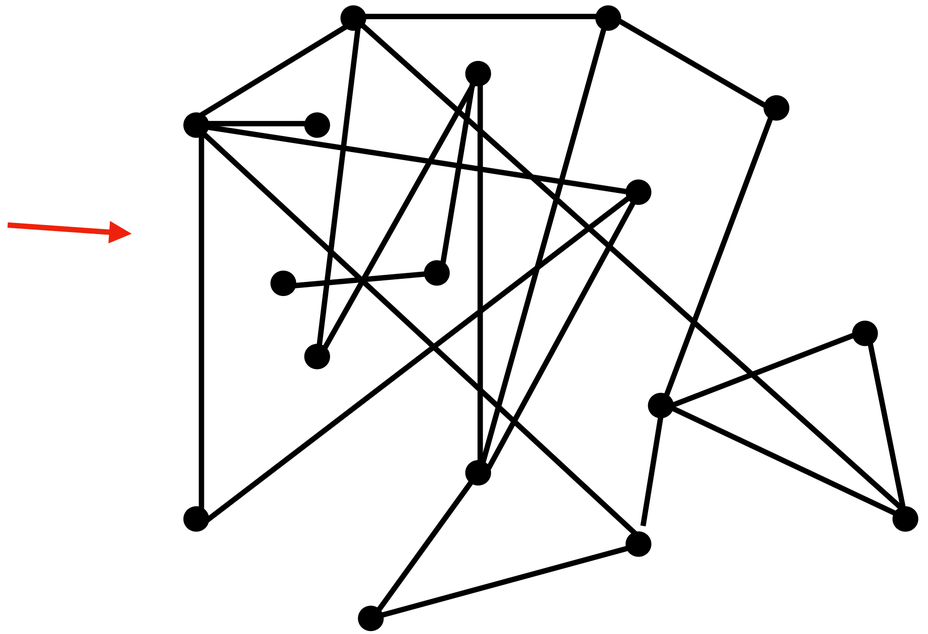
Is it safe to apply $\mathscr{A}$ to x?

Motivating Question

Given specific instance x & average-case algorithm $\mathscr{A}$:

Is it safe to apply $\mathscr{A}$ to x?

Is this graph
random enough for
some average algorithm(s)?

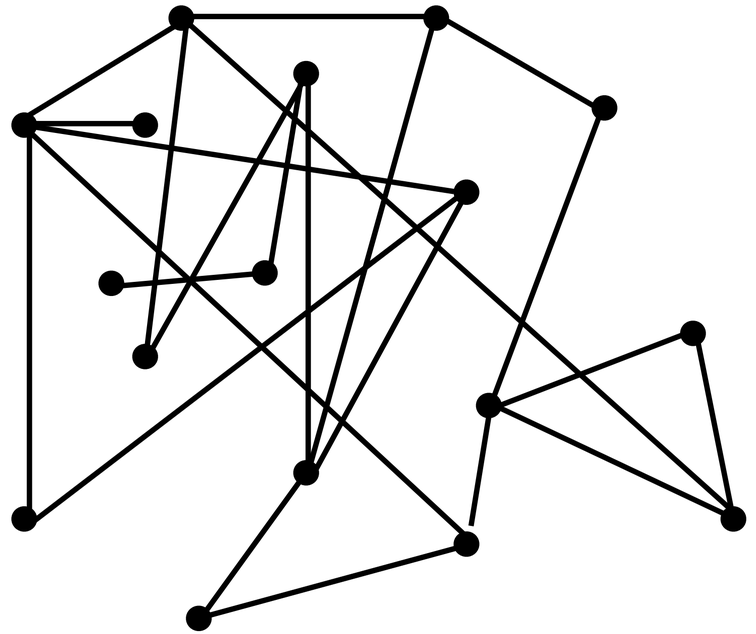


Motivating Question

Given specific instance x & average-case algorithm $\mathscr{A}$:

Is it safe to apply $\mathscr{A}$ to x?

Is this graph
random enough for
some average algorithm(s)?

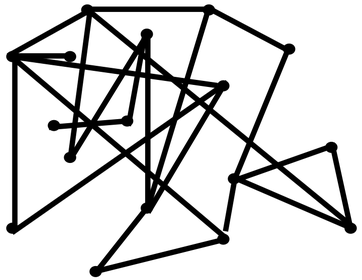


Algorithms over random instances:

e.g. random CSPs, learning theory, planted detection, ….

A common practice

Use a subgraph statistic as certificate of “random-looking”



Estimate k-cliques

Compare to expected count under $\mathcal{G}_{n,p}$

Decision: SAFE/UNSAFE

Expected count: $\binom{n}{k} p^{\binom{k}{2}}$

$\mathcal{G}_{n,p}$ = graph with n vertices and each edge included independently with probability p

Examples: [Chung-Graham-Wilson ’88, Bickel-Chen-Levina ’11, Gleich-Owen ’12]

Old goal: exactly certify the statistic

Accept "SAFE" k-clique count $\approx \binom{n}{k} p^{\binom{k}{2}}$

Reject "UNSAFE" k-clique count $\not\approx \binom{n}{k} p^{\binom{k}{2}}$

A property-testing style guarantee:
separate every good-count graph from every bad-count graph.

But is this the right notion of “safe”?

Old test treats all inputs as worst-case.
However…

- Maybe the graph doesn’t look like $\mathcal{G}_{n,p}$ for another reason.
- Or if it looks typical, maybe can test safety more efficiently.

But is this the right notion of “safe”?

Old test treats all inputs as worst-case.
However…

- Maybe the graph doesn’t look like $\mathcal{G}_{n,p}$ for another reason.
- Or if it looks typical, maybe can test safety more efficiently.

New question:
Can we use that true safe inputs come from $\mathcal{G}_{n,p}$ to test faster?

New goal: changes accept condition

Accept “SAFE”

Typical $\mathcal{G}_{n,p}$ instances

Reject “UNSAFE”

k-clique count $\not\approx \binom{n}{k} p^{\binom{k}{2}}$

Completeness is weakened
Soundness stays worst-case

New goal: changes accept condition

Accept "SAFE" Typical $\mathcal{G}_{n,p}$ instances

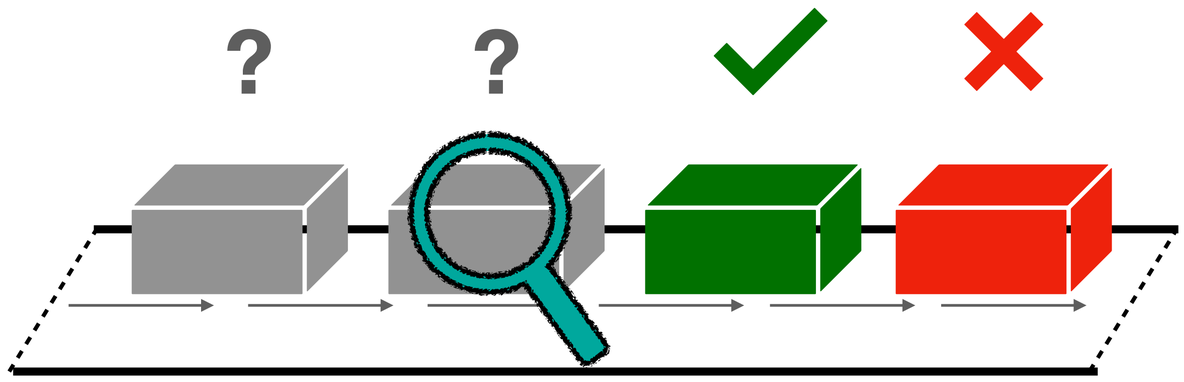
Reject "UNSAFE" k-clique count $\not\approx \binom{n}{k} p^{\binom{k}{2}}$

Goal: construct algorithm with this guarantee.

Hope: more efficient algorithm?

Broader Framework

Quality Control:

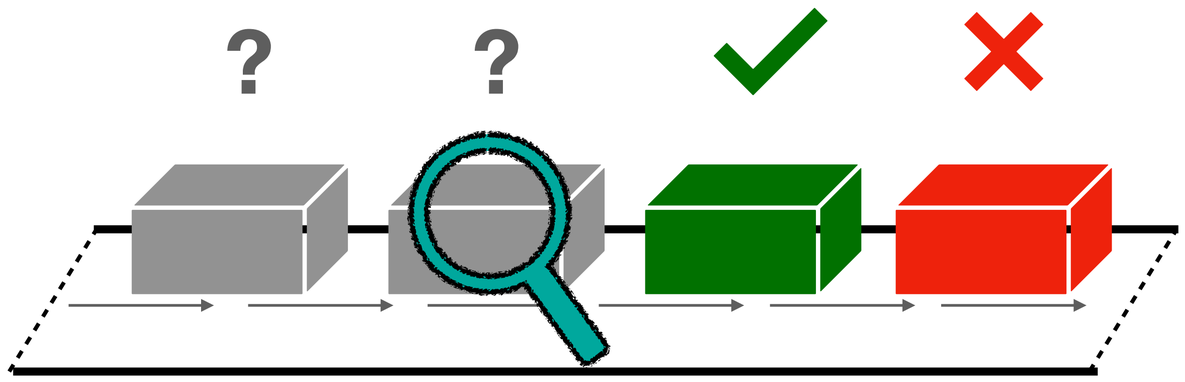


Broader Framework

Quality Control:

HIGH QUALITY (safe) product ⇒ ACCEPT (usually)

LOW QUALITY (unsafe) product ⇒ must REJECT



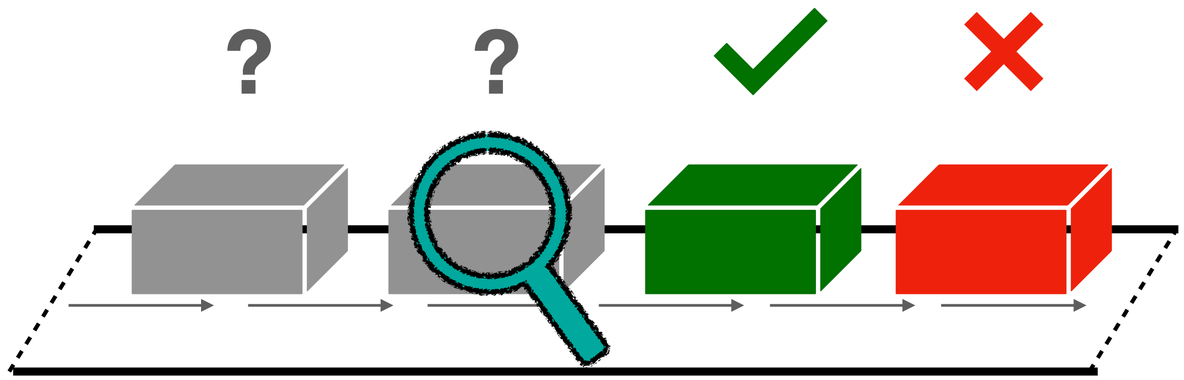
Broader Framework

Quality Control:

HIGH QUALITY (safe) product ⇒ ACCEPT (usually)

LOW QUALITY (unsafe) product ⇒ must REJECT

Suffices if MOST good products accepted.



Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

e.g. $C_k(G) = \#k$-cliques in G

$Q = C_k(G)/\mathbb{E}_{G' \sim \mathcal{G}_{n,p}}[C_k(G')]$:

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

$\mathcal{G}_{n,p}$, $n \geq p^{-(k-1)/2}$

Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

Definition [M., Rubinfeld, Sudan]**:** A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

Definition [M., Rubinfeld, Sudan]**:** A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

"accept (lots of) high-quality products"

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

"reject low-quality products"

Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

Definition [M., Rubinfeld, Sudan]**:** A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Asymmetric definition and one instance

Quality control

Let Q be a function: on input x, $Q(x)$ represents *quality of* x

$Q \approx 1$ means "high quality product"

"Good distribution" D has $Q(x) \in 1 \pm \varepsilon$ whp for $x \sim D$

Definition [M., Rubinfeld, Sudan]**:** A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Study: query complexity and runtime

Comparison to other models

Testable learning [Rubinfeld-Vasilyan ’23]

- Many samples from a distribution
- Accept if learning algorithm safe on distribution, reject otherwise

Testable algorithms [Eden-Rubinfeld-Vasilyan ’26]

- On all graphs with some property, output correct answer
- On all graphs without the property, output correct answer or “reject”

Comparison to other models

Testable learning **[Rubinfeld-Vasilyan '23]**

- Many samples from a distribution
- Accept **if** learning algorithm **safe on distribution**, reject otherwise

Perfect completeness

Testable algorithms **[Eden-Rubinfeld-Vasilyan '26]**

- On **all graphs with some property**, output correct answer
- On all graphs without the property, output correct answer or "reject"

Quality control weakens this to: "accept MOST"

Comparison to other models

Testable learning **[Rubinfeld-Vasilyan '23]**

- Many samples from a distribution
- Accept **if** learning algorithm **safe on distribution**, reject otherwise

Perfect completeness

Testable algorithms **[Eden-Rubinfeld-Vasilyan '26]**

- On **all graphs with some property**, output correct answer
- On all graphs without the property, output correct answer or "reject"

Quality control weakens this to: "accept MOST"

Plus, connections to:
property testing, distribution testing

Examples in the literature

Many quality control examples studied implicitly in previous work:

Subroutine of average-case algorithms (Dyer–Frieze ’89)

Natural proofs (Razborov–Rudich ’97)

Refutation of random CSPs (Feige ’02)

Strong spectral refutation of k-SAT unsatisfiability
(Allen–O’Donnell–Witmer ’15, Raghavendra–Rao–Schramm ’17)

Algorithmic robust statistics: certifiable subgaussianity
(Kothari–Steurer ’17, Hopkins–Li ’18, Diakonikolas–Hopkins–Pensia–Tiegel ’25)

Untrusted randomness (Girish–Raz–Zhan ’23)

Examples in the literature

Many quality control examples studied implicitly in previous work:

Subroutine of average-

Natural proofs (Razborov

Refutation of random C

Strong spectral refutati
(Allen–O'Donnell–Witmer '15

Algorithmic robust statistics: certifiable subgaussianity
(Kothari–Steurer '17, Hopkins–Li '18, Diakonikolas–Hopkins–Pensia–Tiegel '25)

Untrusted randomness (Girish–Raz–Zhan '23)

We formalize the *general framework*

(and focus on sublinear access models)

Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Reminder:

Definition [M., Rubinfeld, Sudan]**:** A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x\sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Erdős–Rényi random graphs:
(each edge present wp p)

Reminder:

Definition [M., Rubinfeld, Sudan]: A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x\sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Erdős–Rényi random graphs:
(each edge present wp p)

$$C_k(G) = \#k\text{-cliques in } G$$

$$Q_k(G) = \frac{C_k(G)}{\mathbb{E}_{G' \sim \mathcal{G}_{n,p}}\left[C_k(G')\right]} = \frac{C_k(G)}{\binom{n}{k} p^{\binom{k}{2}}}$$

Reminder:

Definition [M., Rubinfeld, Sudan]: A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Erdős–Rényi random graphs:
(each edge present wp p)

$$C_k(G) = \#k\text{-cliques in } G$$

$$Q_k(G) = \frac{C_k(G)}{\mathbb{E}_{G' \sim \mathcal{G}_{n,p}}\left[C_k(G')\right]} = \frac{C_k(G)}{\binom{n}{k} p^{\binom{k}{2}}}$$

Reminder:

Definition [M., Rubinfeld, Sudan]: A is a (D, Q)-quality control algorithm if

Completeness: $\mathbb{P}_{x \sim D}\left[A(x) \text{ accepts}\right] = 1 - o(1)$

Soundness: $\forall x$ with $Q(x) \notin 1 \pm \varepsilon$, $A(x)$ rejects with probability $\geq 2/3$

$Q_k(\mathcal{G}_{n,p}) \approx 1$ whp for $n \geq p^{-ck}$, some c

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Erdős–Rényi random graphs:
(each edge present wp p)

$C_k(G) = \#k$-cliques in G

$$Q_k(G) = \frac{C_k(G)}{\mathbb{E}_{G' \sim \mathcal{G}_{n,p}}\left[C_k(G')\right]} = \frac{C_k(G)}{\binom{n}{k} p^{\binom{k}{2}}}$$

Does the k-clique count match $\mathcal{G}_{n,p}$?

$Q_k(\mathcal{G}_{n,p}) \approx 1$ whp for $n \geq p^{-ck}$, some c

Our setting (+ notation)

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Erdős–Rényi random graphs:
(each edge present wp p)

Does the k-clique count match $\mathcal{G}_{n,p}$?

Why study clique/motif counts?

Fundamental task in network science

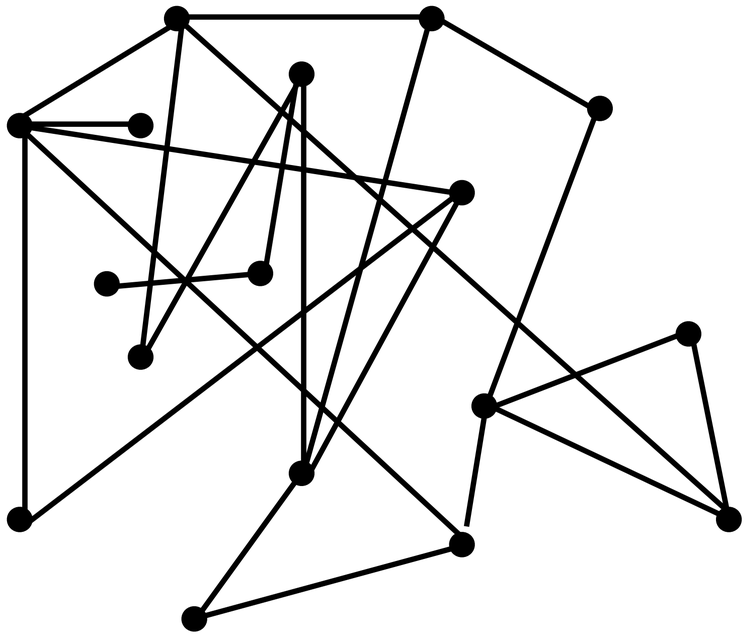
Connections to:
Quasirandomness,
Graph limits,
Moments of a graph*, …

Q_k *expensive to compute/verify*

* Bickel, Chen, Levina 2011. The method of moments and degree distributions for network models.

Our setting

Sublinear graph access



Queries:

Is (u, v) an edge?

Adjacency matrix query

What is the i-th neighbor of u?

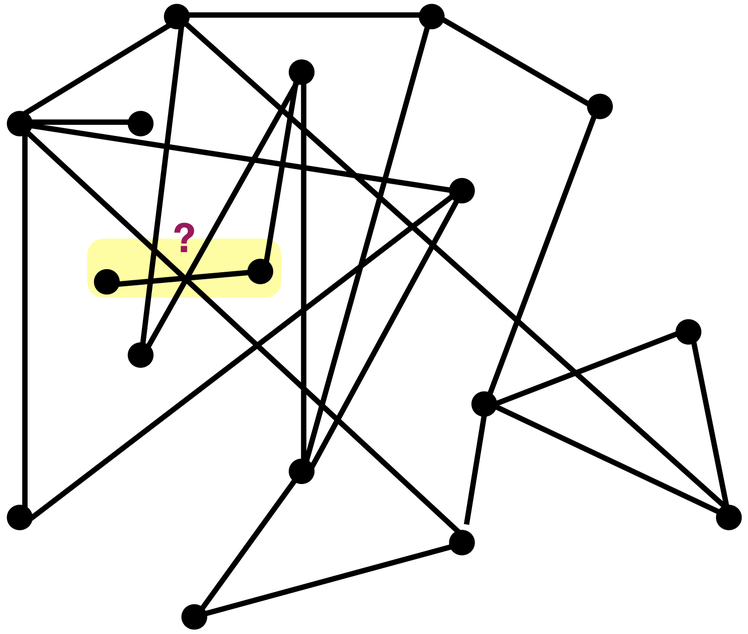
Adjacency list query

What is the degree of u?

Degree query

Our setting

Sublinear graph access



Queries:

Only need these

Is (u, v) an edge?

Adjacency matrix query

What is the i-th neighbor of u?

Adjacency list query

What is the degree of u?

Degree query

Comparison to worst-case setting

Is $Q_k(G) \in 1 \pm \varepsilon$?

$1/p^{\Theta(k^2)}$ queries/time
for worst-case graphs
[Eden, Ron, Seshadhri 2018]

Comparison to worst-case setting

Is $Q_k(G) \in 1 \pm \varepsilon$?

$1/p^{\Theta(k^2)}$ queries/time
for worst-case graphs
[Eden, Ron, Seshadhri 2018]

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Theorem [M., Rubinfeld, Sudan]
$1/p^{\Theta(k)}$ queries/time*

*Tight up to universal constants in the exponent

Comparison to worst-case setting

Is $Q_k(G) \in 1 \pm \varepsilon$?

$1/p^{\Theta(k^2)}$ queries/time
for worst-case graphs
[Eden, Ron, Seshadhri 2018]

$(\mathcal{G}_{n,p}, Q_k)$-quality control

Theorem [M., Rubinfeld, Sudan]
$1/p^{\Theta(k)}$ queries/time*

$\exists\ k, p, c$ such that for $n \geq p^{-ck}$
[ERS18] requires $\Omega(n^k)$ queries/time, and
we require $o(n)$.

*Tight up to universal constants in the exponent

Main results

Theorem: [M., Rubinfeld, Sudan]

1. $(\mathcal{G}_{n,p}, Q_k)$-quality control in $O\left((1/p)^{ck}\right)$ time and queries

1'. Tight up to c

Main results

Theorem: [M., Rubinfeld, Sudan]

1. $(\mathcal{G}_{n,p}, Q_k)$-quality control in $O\left((1/p)^{ck}\right)$ time and queries

1'. Tight up to c

2. (General version of 1) Quality control for count of motif with max degree Δ compared to $\mathcal{G}_{n,p}$: $O\left((1/p)^{c\Delta}\right)$ time and queries

2'. Tight up to c

Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

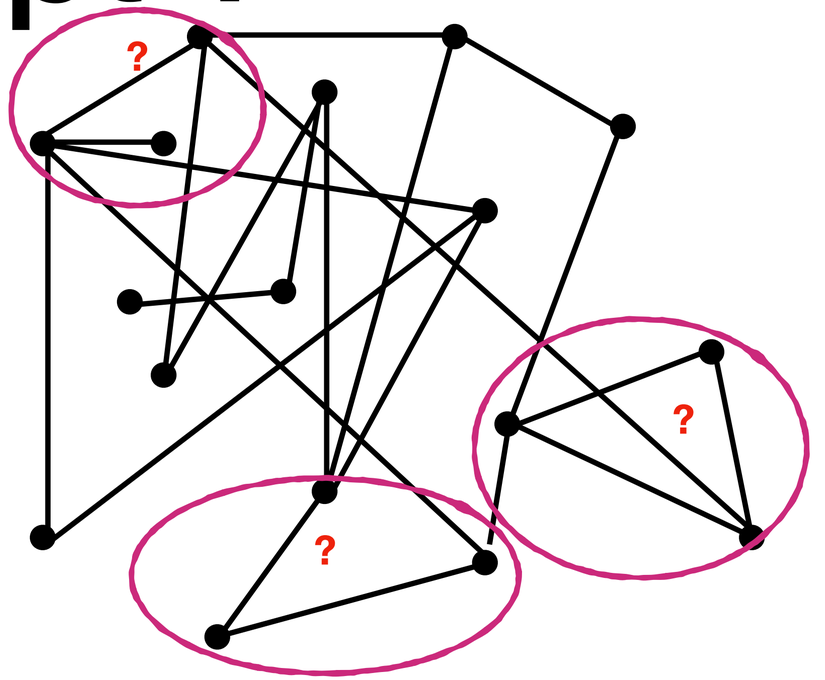
Algorithm: Attempt 1

Repeat $t = ??$ times:

Pick random set of k vertices

Check if k-clique

Accept if empirical frequency $\approx p^{\binom{k}{2}} =: M_k$



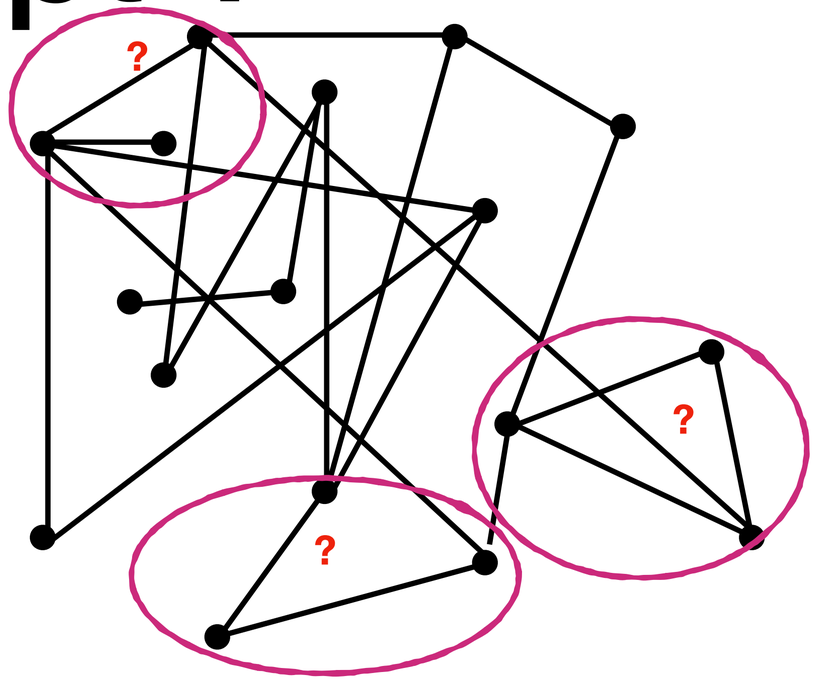
Algorithm: Attempt 1

Repeat $t = ??$ times:

Pick random set of k vertices

Check if k-clique

Accept if empirical frequency $\approx p^{\binom{k}{2}} =: M_k$



Need $(1/p)^{\Theta(k^2)}$ time just to ***FIND*** k-clique!

$$t = \Omega(1/M_k) = (1/p)^{\Omega(k^2)}$$

Algorithm: Attempt 2

Pick random set $S \subseteq [n]$ with $|S| = s = ??$

Count $C_k(S) = \#\ k$-cliques in subgraph on S

Reject if $C_k(S) \not\approx \binom{s}{k} p^{\binom{k}{2}}$

Else Accept.

Algorithm: Attempt 2

Pick random set $S \subseteq [n]$ with $|S| = s = ??$

Count $C_k(S) = \#\ k$-cliques in subgraph on S

Reject if $C_k(S) \not\approx \binom{s}{k} p^{\binom{k}{2}}$

Else Accept.

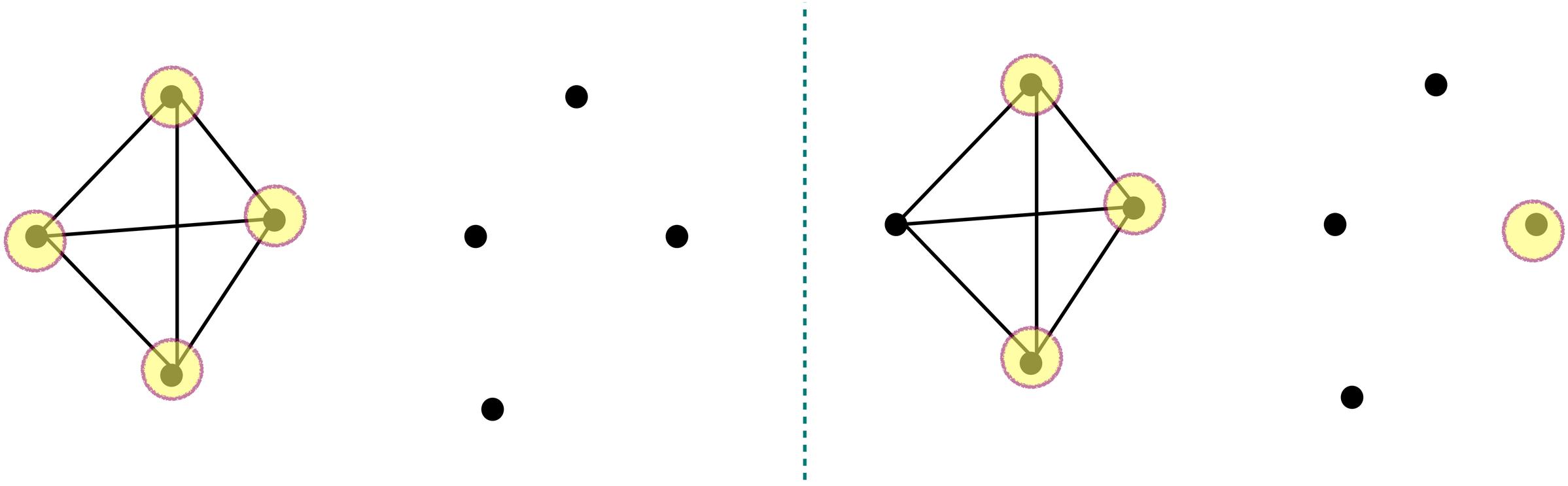
Completeness okay for $s = (1/p)^{O(k)}$, since $\binom{s}{k} p^{\binom{k}{2}} \gg 1$.

Soundness / analysis?

Variance issue

In worst-case graphs, changing one vertex of S can impact $C_k(S)$ by $(|S|/2)^{k-1}$

Will need $|S| = 1/p^{\Omega(k^2)}$ for good concentration



What we want

How can quality control help?

Allowed to “filter”, as long as accept $G \sim \mathscr{G}_{n,p}$ and reject low-quality G

What we want

How can quality control help?

Allowed to “filter”, as long as accept $G \sim \mathcal{G}_{n,p}$ and reject low-quality G

Core principle of quality control: Composability

Recursively perform (D, P)-quality control for (D, Q)-quality control

Composability

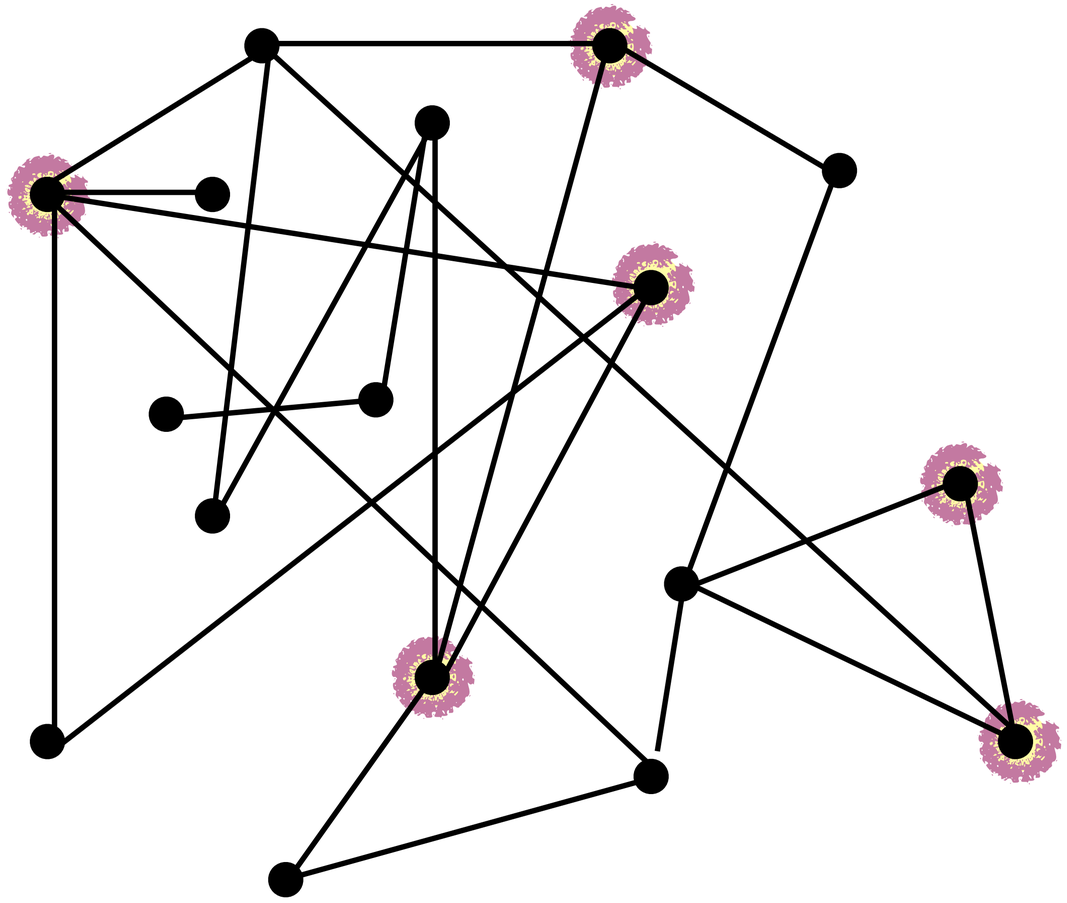
What property do we track/control?

The counts of all ℓ-cliques, $\ell \leq k$ ($P = \#\ \ell$-cliques)

Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

Final (query optimal) algorithm



Pick random set $S \subseteq [n]$,
$|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

Composability with cliques

What property do we track/control?

The counts of all ℓ-cliques, $\ell \leq k$ ($P = \#\ \ell$-cliques)

More concretely, we want to find:

Property $P(\ell)$: An inductive, testable property on ℓ-cliques

$\Rightarrow$

Can efficiently count $(\ell + 1)$-cliques

and see if Property $P(\ell + 1)$ holds

Composability with cliques

Property $P(\ell)$: "Most" subgraphs of G of size s have

$$\approx \binom{s}{\ell} p^{\binom{\ell}{2}} \ell\text{-cliques}$$

Most = $1 - \exp(-\Omega(s))$ **fraction**

Composability with cliques

Property $P(\ell)$: "Most" subgraphs of G of size s have

$$\approx \binom{s}{\ell} p^{\binom{\ell}{2}} \ell\text{-cliques}$$

$$\Rightarrow$$

With high probability, S of size s has $C_{\ell+1}(S) \propto C_{\ell+1}(G)$

Most = $1 - \exp(-\Omega(s))$ **fraction**

Composability with cliques

Property $P(\ell)$: "Most" subgraphs of G of size s have

$$\approx \binom{s}{\ell} p^{\binom{\ell}{2}} \ell\text{-cliques}$$

$$\Rightarrow$$

With high probability, S of size s has $C_{\ell+1}(S) \propto C_{\ell+1}(G)$

$$\Rightarrow$$

Can efficiently count $(\ell+1)$-cliques

and see if Property $P(\ell+1)$ holds

Most = $1 - \exp(-\Omega(s))$ **fraction**

Composability with cliques

Property $P(\ell)$: "Most" subgraphs of G of size s have

$$\approx \binom{s}{\ell} p^{\binom{\ell}{2}} \ell\text{-cliques}$$

$\Rightarrow$

With high probability, S of size s has $C_{\ell+1}(S) \propto C_{\ell+1}(G)$

$\Rightarrow$

Can efficiently count $(\ell+1)$-cliques

and see if Property $P(\ell+1)$ holds

Most = $1 - \exp(-\Omega(s))$ **fraction**

Exponentially robust quasirandomness

Definition: (Exponentially robust quasirandomness) **[M., Rubinfeld, Sudan]**

$(\alpha, \varepsilon, s_0)$-exponentially robustly quasirandom wrt k-cliques if $\forall s \geq s_0$:

$$\mathbb{P}_{\substack{S \subseteq [n] \\ |S| = s}} \left[C_k(S) \in (1 \pm \varepsilon) \binom{s}{k} p^{\binom{k}{2}} \right] \geq 1 - \exp(-\alpha s)$$

Exponentially robust quasirandomness

Definition: (Exponentially robust quasirandomness) **[M., Rubinfeld, Sudan]**

$(\alpha, \varepsilon, s_0)$-exponentially robustly quasirandom wrt k-cliques if $\forall s \geq s_0$:

$$\mathbb{P}_{\substack{S \subseteq [n] \\ |S| = s}} \left[C_k(S) \in (1 \pm \varepsilon) \binom{s}{k} p^{\binom{k}{2}} \right] \geq 1 - \exp(-\alpha s)$$

We prove:

- Can inductively verify exponentially robust quasirandomness efficiently
- If exponentially robustly quasirandom for ℓ-cliques, can efficiently count $(\ell + 1)$-cliques

Remark: Graph quasirandomness

Graph quasirandomness: *global*, *strong*, implies # k-cliques matches $\mathcal{G}_{n,p}$

Not efficiently verifiable.

(e.g. need “every vertex”, “every pair of vertices”, “every subgraph” properties)

Remark: Graph quasirandomness

Graph quasirandomness: *global*, *strong*, implies # k-cliques matches $\mathcal{G}_{n,p}$

Not efficiently verifiable.

(e.g. need “every vertex”, “every pair of vertices”, “every subgraph” properties)

Robust notion of graph quasirandomness: implies # k-cliques matches $\mathcal{G}_{n,p}$ WHP

Efficiently verifiable.

Back to: final algorithm

Pick random set $S \subseteq [n]$,
$|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

Back to: final algorithm

Keep graphs when exponentially robustly **quasirandom wrt ℓ-cliques**.

Reject graph **when not**.

Pick random set $S \subseteq [n]$, $|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

Back to: final algorithm

Keep graphs when exponentially robustly **quasirandom wrt ℓ-cliques**.

Reject graph **when not**.

Therefore easier to count $(\ell+1)$-cliques

Pick random set $S \subseteq [n]$, $|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

Agenda

- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

So far…

$s^2 = (1/p)^{O(k)}$ queries

Beats $(1/p)^{\Omega(k^2)}$ worst-case lower bound

Algorithm:

Pick a random set $S \subseteq [n]$ with $|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

So far…

$s^2 = (1/p)^{O(k)}$ queries

Beats $(1/p)^{\Omega(k^2)}$ worst-case lower bound

$s^k = (1/p)^{O(k^2)}$ runtime
(to count cliques in subgraph)

Can we beat $(1/p)^{\Omega(k^2)}$ worst-case lower bound?

Algorithm:

Pick a random set $S \subseteq [n]$ with $|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

So far…

Need:

Most size-s subgraphs of $\mathcal{G}_{n,p}$

$\Rightarrow$

Property P

$\Rightarrow$

All $C_\ell(S) \approx \binom{s}{\ell} p^{\binom{\ell}{2}}$,

$\ell = 2,3,\ldots,k$

Algorithm:

Pick a random set $S \subseteq [n]$ with $|S| = s = 1/p^{O(k)}$

For $\ell = 2,3,\ldots,k$:

Count $C_\ell(S) = \#\ \ell$-cliques in S

Reject if $C_\ell(S) \not\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$

Else Accept.

So far…

Need:

Most size-s subgraphs of $\mathcal{G}_{n,p}$

$\Rightarrow$

Property P

$\Rightarrow$

All $C_\ell(S) \approx \binom{s}{\ell} p^{\binom{\ell}{2}}$,

$\ell = 2,3,\ldots,k$

Will be:
Another notion of
graph quasirandomness

Efficient approximate counting

Theorem: [Thomason ’87]

Let G be an s-vertex graph such that all $\deg(u) \approx ps$ and $\text{codeg}(u, v) \approx p^2 s$.

codeg(u, v) = # shared neighbors of u and v

Efficient approximate counting

Theorem: [Thomason '87]

Let G be an s-vertex graph such that all $\deg(u) \approx ps$ and $\text{codeg}(u, v) \approx p^2 s$.

Then, for all $\ell = 2,3,\ldots,k$, the number of ℓ-cliques in G is $\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$.

codeg(u, v) = # shared neighbors of u and v

Efficient approximate counting

Theorem: [Thomason '87]

Let G be an s-vertex graph such that all $\deg(u) \approx ps$ and $\text{codeg}(u, v) \approx p^2 s$.

Then, for all $\ell = 2,3,\ldots,k$, the number of ℓ-cliques in G is $\approx \binom{s}{\ell} p^{\binom{\ell}{2}}$.

Relies on:
"jumbledness"

codeg(u, v) = # shared neighbors of u and v

Final (runtime optimal) algorithm

Pick random set $S \subseteq [n]$, $|S| = s = 1/p^{O(k)}$

Reject if any $\deg_S(u) \not\approx ps$ or $\text{codeg}_S(u, v) \not\approx p^2 s$

Else Accept.

Final (runtime optimal) algorithm

Pick random set $S \subseteq [n]$, $|S| = s = 1/p^{O(k)}$

Reject if any $\deg_S(u) \not\approx ps$ or $\operatorname{codeg}_S(u,v) \not\approx p^2 s$

Else Accept.

Runs quality control for quasirandomness on S, $O(s^3)$ time

Final (runtime optimal) algorithm

Pick random set $S \subseteq [n]$, $|S| = s = 1/p^{O(k)}$

Reject if any $\deg_S(u) \not\approx ps$ or $\mathrm{codeg}_S(u, v) \not\approx p^2 s$

Else Accept.

Runs quality control for quasirandomness on S, $O(s^3)$ time

Implies all ℓ-clique count matches $\mathcal{G}_{n,p}$, $\ell \leq k$

What we've proven

Theorem:

$(\mathcal{G}_{n,p}, Q_k)$-quality control in $O\left((1/p)^{ck}\right)$ time and queries.

Super-polynomial improvement over the $1/p^{\Theta(k^2)}$ bound for worst-case graphs.

$\exists\ k, p, c$ such that for $n \geq p^{-ck}$
[ERS18] requires $\Omega(n^k)$ queries/time, and
we require $o(n)$.

We also prove: lower bounds, results for general motifs and other graph distributions.

Agenda

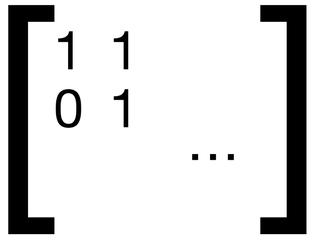
- Quality control
- Our setting and results
- Initial algorithms
- Query-optimal algorithm
- Runtime-optimal algorithm
- Open problems

Open problems

Different distributions
and quality functions

Open problems

Different distributions
and quality functions



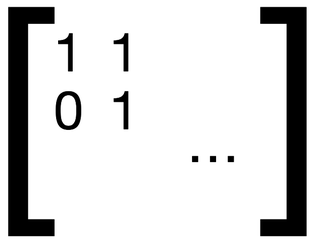
& counts of sub matrices

Other distributions $\mathscr{D}$

and quality functions ρ

Open problems

Different distributions
and quality functions



& counts of sub matrices

Other distributions $\mathscr{D}$

and quality functions ρ

When does quality control yield
more efficient algorithms?

Open problems

Different distributions and quality functions

$$\begin{bmatrix} 1 & 1 & \\ 0 & 1 & \\ & & \ddots \end{bmatrix}$$ & counts of sub matrices

Other distributions $\mathscr{D}$

and quality functions ρ

When does quality control yield more efficient algorithms?

Characterizations of quality control

Open problems

Different distributions and quality functions

$$\begin{bmatrix} 1 & 1 & \\ 0 & 1 & \\ & \dots & \end{bmatrix}$$ & counts of sub matrices

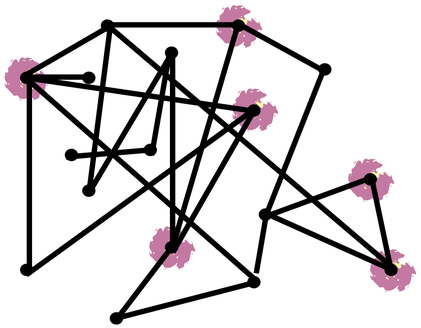
Other distributions $\mathscr{D}$

and quality functions ρ

When does quality control yield more efficient algorithms?

Characterizations of quality control

Canonical tester?



Open problems

Different distributions and quality functions

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \\ & \cdots \end{bmatrix}$$ & counts of sub matrices

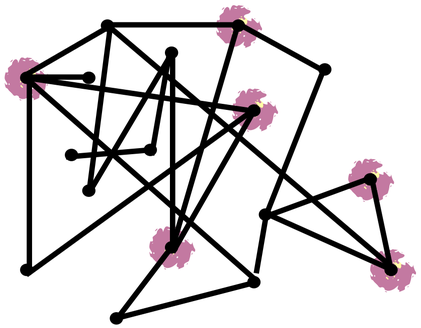
Other distributions $\mathscr{D}$

and quality functions ρ

When does quality control yield more efficient algorithms?

Characterizations of quality control

Canonical tester?



Combinatorial characterizations?

Conclusion

Quality control: Accept $x \sim D$ whp and reject when $Q(x) \not\approx 1$

k-cliques: $1/p^{\Theta(k)}$ queries/runtime

General motif H: $1/p^{\Theta(\Delta(H))}$ **where $\Delta(H)$ is max degree**

Key tools: 1) Composability
2) A robust version of graph “quasirandomness”

Future directions: Other distributions and quality functions; characterizations

Thank you!