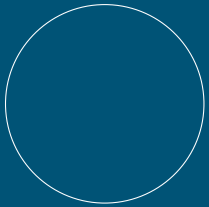
A Simple Quantum Sketch With Applications to Graph Algorithms

John Kallaugher[1]

[1] Quantum Algorithms and Applications Collaboratory
Sandia National Labs

Sampling

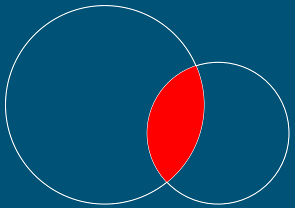
Suppose we have a set $S \subseteq U$, and we want a small sketch of it that solves:



Sampling

Suppose we have a set $S \subseteq U$, and we want a small sketch of it that solves:

Cardinality Given $P \subseteq U$, how large is $S \cap P$? (to e.g. $\varepsilon|S| = O(|S|)$ error)

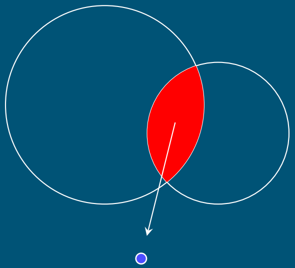


Sampling

Suppose we have a set $S \subseteq U$, and we want a small sketch of it that solves:

Cardinality Given $P \subseteq U$, how large is $S \cap P$? (to e.g. $\varepsilon|S| = O(|S|)$ error)

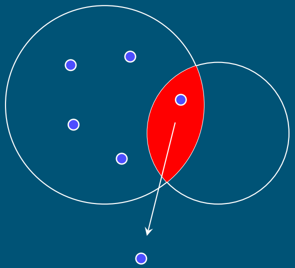
Sampling If $|P \cap U| = \Omega(|S|)$, return a random element of $|P \cap U|$ with prob $\Omega(1)$.



Sampling

Suppose we have a set $S \subseteq U$, and we want a small sketch of it that solves:

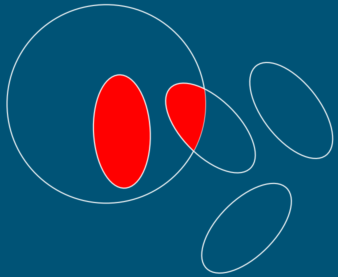
- **Cardinality** Given $P \subseteq U$, how large is $S \cap P$? (to e.g. $\varepsilon|S| = O(|S|)$ error)
- **Sampling** If $|P \cap U| = \Omega(|S|)$, return a random element of $|P \cap U|$ with prob $\Omega(1)$.



Simple solution: uniformly sample $O(1)$ elements from S, using $O(\log|U|)$ bits.

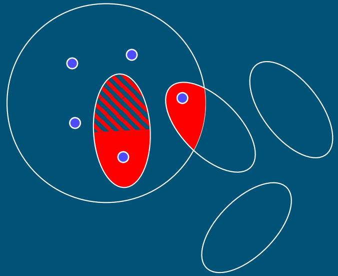
Sampling Pairs

What if P is a set of (disjoint) *pairs* from U instead?



Sampling Pairs

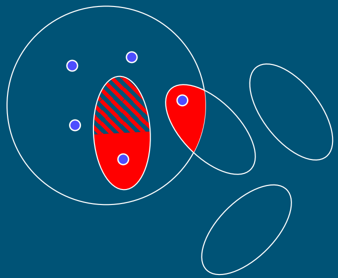
What if P is a set of (disjoint) *pairs* from U instead?



- Probability of a random sample hitting both elements of a pair is much lower.

Sampling Pairs

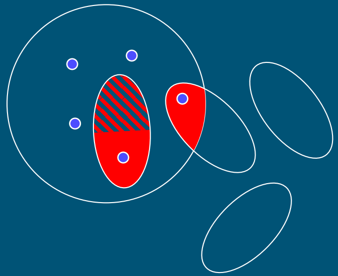
What if P is a set of (disjoint) *pairs* from U instead?



- Probability of a random sample hitting both elements of a pair is much lower.
- Need $\Theta\left(\sqrt{|S|}\right)$ samples even with $|\{p \in P : p \subseteq S\}| = \Omega(|P|)$.

Sampling Pairs

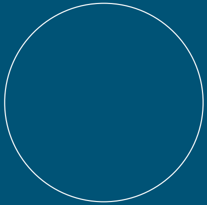
What if P is a set of (disjoint) *pairs* from U instead?



- Probability of a random sample hitting both elements of a pair is much lower.
- Need $\Theta\left(\sqrt{|S|}\right)$ samples even with $|\{p \in P : p \subseteq S\}| = \Omega(|P|)$.
- This is optimal up to a log factor by reduction to Boolean Hidden Matching [Gavinsky, Kempe, Kerenidis, Raz, de Wolf].

Quantum Sketch

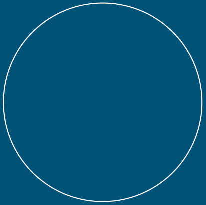
We can do better with a *quantum* sketch (inspired by a protocol for BHM).



$$\Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

Quantum Sketch

We can do better with a *quantum* sketch (inspired by a protocol for BHM).

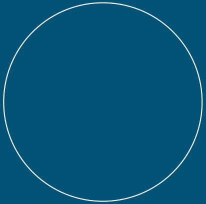


$$\Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

With $O(\log|U|)$ qubits:

Quantum Sketch

We can do better with a *quantum* sketch (inspired by a protocol for BHM).



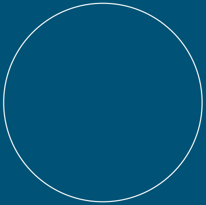
$$\Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

With $O(\log|U|)$ qubits:

Cardinality Estimate $|\{p \in P : p \subseteq S\}|$ to $\varepsilon|S| = O(|S|)$ error.

Quantum Sketch

We can do better with a *quantum* sketch (inspired by a protocol for BHM).



$$\Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

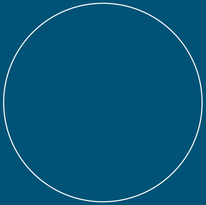
With $O(\log |U|)$ qubits:

Cardinality Estimate $|\{p \in P : p \subseteq S\}|$ to $\varepsilon |S| = O(|S|)$ error.

Sampling If $|\{p \in P : p \subseteq S\}| = \Omega(|S|)$, return a random element of $\{p \in P : p \subseteq S\}$ with probability $\Omega(1)$.

Quantum Sketch

We can do better with a *quantum* sketch (inspired by a protocol for BHM).



$$\Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

With $O(\log|U|)$ qubits:

Cardinality Estimate $|\{p \in P : p \subseteq S\}|$ to $\varepsilon|S| = O(|S|)$ error.

Sampling If $|\{p \in P : p \subseteq S\}| = \Omega(|S|)$, return a random element of $\{p \in P : p \subseteq S\}$ with probability $\Omega(1)$.

- Otherwise: can return p s.t. $|p \cap S| = 1$, but with $\frac{1}{2}$ chance of -1 label.

Constructing the Quantum Sketch

Two Quantum Primitives

Superposition

A superposition $\sum_{x \in U} \alpha_x |x\rangle$ over U is a unit-length vector in $\mathbb{C}^U$.

Write $|x\rangle$ for the basis element corresponding to $x \in U$.

Two Quantum Primitives

Superposition

A superposition $\sum_{x \in U} \alpha_x |x\rangle$ over U is a unit-length vector in $\mathbb{C}^U$.

Write $|x\rangle$ for the basis element corresponding to $x \in U$.

- Can express a superposition over U with $\log|U|$ qubits.

Two Quantum Primitives

Superposition

A superposition $\sum_{x \in U} \alpha_x |x\rangle$ over U is a unit-length vector in $\mathbb{C}^U$.

Write $|x\rangle$ for the basis element corresponding to $x \in U$.

- Can express a superposition over U with $\log|U|$ qubits.

Projective Measurement

Using labeled projectors (Λ_i) s.t. $\sum_i \Lambda_i = I$, measure state ψ. With probability $\|\Lambda_i \psi\|_2^2$, get result i and transform ψ to $\Lambda_i \psi / \|\Lambda_i \psi\|_2$.

Constructing the Quantum Sketch

The sketch will be O(1) copies of the *uniform superposition* over S.

$$S \Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

Constructing the Quantum Sketch

The sketch will be O(1) copies of the *uniform superposition* over S.

$$S \Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

To query: measure each copy with projectors onto $|x\rangle + |y\rangle$ and $|x\rangle - |y\rangle$ for each $\{x, y\} \in P$.

Constructing the Quantum Sketch

The sketch will be O(1) copies of the *uniform superposition* over S.

$$S \Rightarrow \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

To query: measure each copy with projectors onto $|x\rangle + |y\rangle$ and $|x\rangle - |y\rangle$ for each $\{x, y\} \in P$.

- P being *disjoint* pairs is necessary here as it makes these projectors orthogonal.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.
- $\{x, y\} \cap S = \emptyset$: neither of $|x\rangle \pm |y\rangle$ will ever be returned.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.
- $\{x, y\} \cap S = \emptyset$: neither of $|x\rangle \pm |y\rangle$ will ever be returned.

So with a large enough constant number of copies we can 1) estimate $|\{p \in P : p \subseteq S\}|$ and 2) sample, provided $|p \in P : p \subseteq S| = \Omega(|S|)$.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.
- $\{x, y\} \cap S = \emptyset$: neither of $|x\rangle \pm |y\rangle$ will ever be returned.

So with a large enough constant number of copies we can 1) estimate $|\{p \in P : p \subseteq S\}|$ and 2) sample, provided $|p \in P : p \subseteq S| = \Omega(|S|)$.

Cardinality $\sum_{\{x,y\}\in P}(\mathbb{P}\left[|x\rangle + |y\rangle\right] - \mathbb{P}\left[|x\rangle - |y\rangle\right]) = 2|\{p \in P : p \subseteq S\}|$.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.
- $\{x, y\} \cap S = \emptyset$: neither of $|x\rangle \pm |y\rangle$ will ever be returned.

So with a large enough constant number of copies we can 1) estimate $|\{p \in P : p \subseteq S\}|$ and 2) sample, provided $|p \in P : p \subseteq S| = \Omega(|S|)$.

Cardinality $\sum_{\{x,y\}\in P}(\mathbb{P}[|x\rangle + |y\rangle] - \mathbb{P}[|x\rangle - |y\rangle]) = 2|\{p \in P : p \subseteq S\}|$.

Sampling Return (x, y) with sign $(-1)^b$ on seeing $|x\rangle + (-1)^b|y\rangle$.

Measurement Outcomes

- If $\{x, y\} \subseteq S$: $\frac{2}{|S|}$ chance of returning $|x\rangle + |y\rangle$.
- If $|\{x, y\} \cap S| = 1$: $\frac{1}{2|S|}$ chance of returning each of $|x\rangle \pm |y\rangle$.
- $\{x, y\} \cap S = \emptyset$: neither of $|x\rangle \pm |y\rangle$ will ever be returned.

So with a large enough constant number of copies we can 1) estimate $|\{p \in P : p \subseteq S\}|$ and 2) sample, provided $|p \in P : p \subseteq S| = \Omega(|S|)$.

Cardinality $\sum_{\{x,y\} \in P} (\mathbb{P}[|x\rangle + |y\rangle] - \mathbb{P}[|x\rangle - |y\rangle]) = 2|\{p \in P : p \subseteq S\}|$.

Sampling Return (x, y) with sign $(-1)^b$ on seeing $|x\rangle + (-1)^b|y\rangle$.

- So when we return p with size-1 overlap, it comes with a random sign.

Application: Quantum Advantage for Counting Triangles in the Stream

Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

- Graph problems where the input is received *one edge at a time*.

Graph Streaming

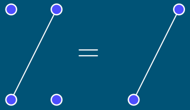
Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

- Graph problems where the input is received *one edge at a time*.

Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

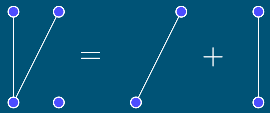
- Graph problems where the input is received *one edge at a time*.



Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

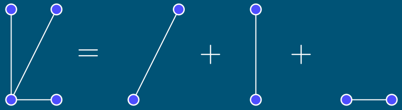
- Graph problems where the input is received *one edge at a time*.



Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

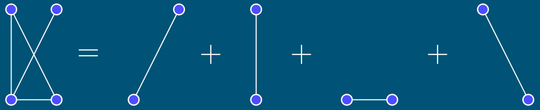
- Graph problems where the input is received *one edge at a time*.



Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

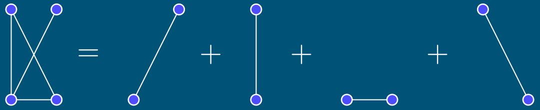
- Graph problems where the input is received *one edge at a time*.



Graph Streaming

Remainder of this talk: two applications of the sketch to problems in *graph* streaming.

- Graph problems where the input is received *one edge at a time*.



We will be concerned with *space complexity*.

Quantum-Classical Separations

Are there streaming problems that require asymptotically fewer qubits than bits?

Quantum-Classical Separations

Are there streaming problems that require asymptotically fewer qubits than bits?

- [Le Gall '06] Can require exponentially less space than classical algorithms (but for an "unnatural" problem).

Quantum-Classical Separations

Are there streaming problems that require asymptotically fewer qubits than bits?

- [Le Gall '06] Can require exponentially less space than classical algorithms (but for an "unnatural" problem).
- [Montanaro '16] Quantum advantage for moment estimation with many ($\omega(1)$) passes over the input.

Quantum-Classical Separations

Are there streaming problems that require asymptotically fewer qubits than bits?

- [Le Gall '06] Can require exponentially less space than classical algorithms (but for an "unnatural" problem).
- [Montanaro '16] Quantum advantage for moment estimation with many ($\omega(1)$) passes over the input.

With this sketch we obtain quantum advantage for single-pass streaming problems of independent interest.

Triangle Counting

Given a graph one edge at a time, count the number of three-cliques.

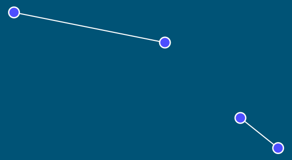
Triangle Counting

Given a graph one edge at a time, count the number of three-cliques.

$$T = 0$$

Triangle Counting

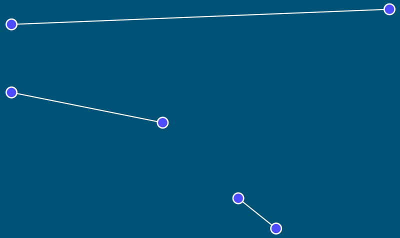
Given a graph one edge at a time, count the number of three-cliques.



$$T = 0$$

Triangle Counting

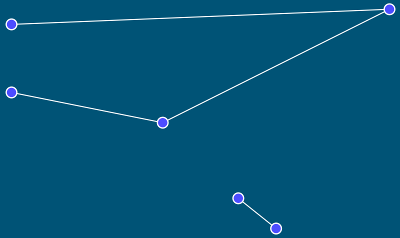
Given a graph one edge at a time, count the number of three-cliques.



$$T = 0$$

Triangle Counting

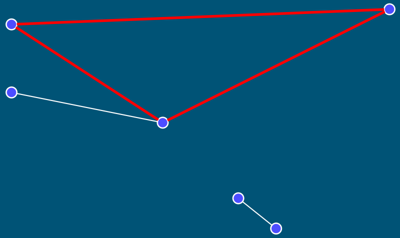
Given a graph one edge at a time, count the number of three-cliques.



$T = 0$

Triangle Counting

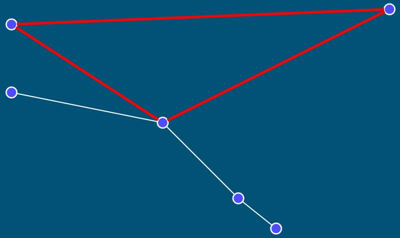
Given a graph one edge at a time, count the number of three-cliques.



$T = 1$

Triangle Counting

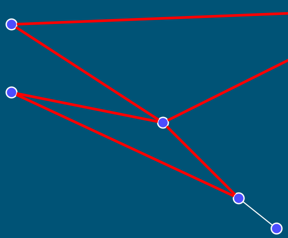
Given a graph one edge at a time, count the number of three-cliques.



$T = 1$

Triangle Counting

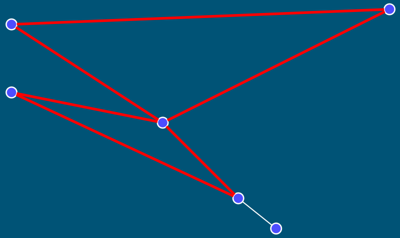
Given a graph one edge at a time, count the number of three-cliques.



$$T = 2$$

Triangle Counting

Given a graph one edge at a time, count the number of three-cliques.

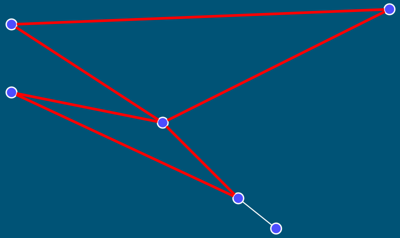


$$T = 2$$

- Applications in social science, spam detection, etc.

Triangle Counting

Given a graph one edge at a time, count the number of three-cliques.

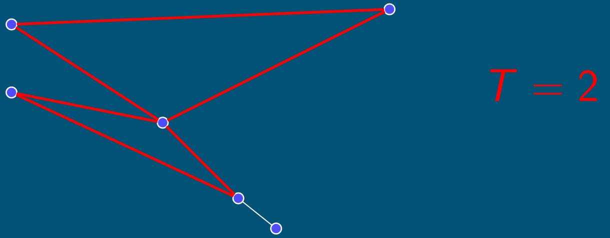


$$T = 2$$

- Applications in social science, spam detection, etc.
- The simplest graph counting problem that requires *non-local* information.

Triangle Counting

Given a graph one edge at a time, count the number of three-cliques.



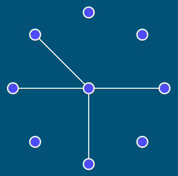
- Applications in social science, spam detection, etc.
- The simplest graph counting problem that requires *non-local* information.
- For our purposes: assume $\Omega(m)$ edge-disjoint triangles in a $\Theta(m)$-edge graph.

Hard Case

One hard case is an m-star with triangles (possibly) completed by a matching.

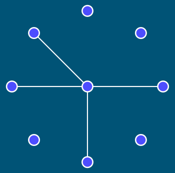
Hard Case

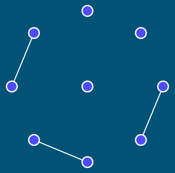
One hard case is an m-star with triangles (possibly) completed by a matching.



Hard Case

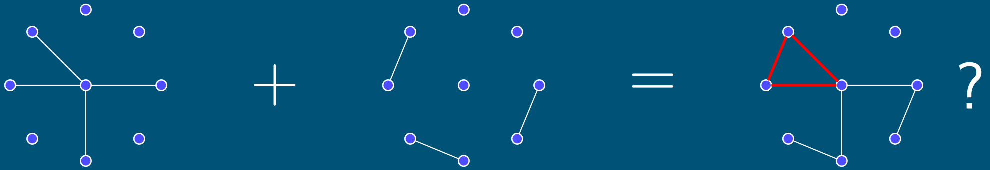
One hard case is an m-star with triangles (possibly) completed by a matching.





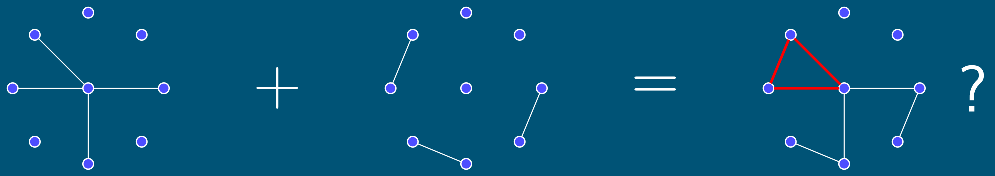
Hard Case

One hard case is an m-star with triangles (possibly) completed by a matching.



Hard Case

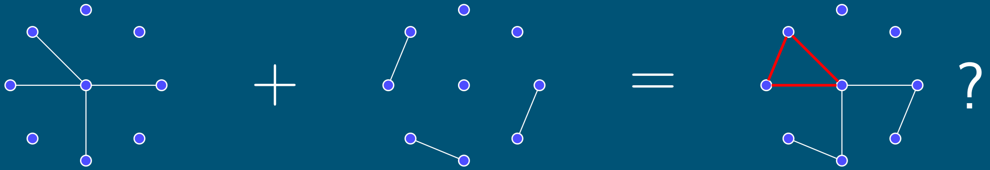
One hard case is an m-star with triangles (possibly) completed by a matching.



- Intuitively: if we keep k star edges, we have a $\sim T \times \left(\frac{k}{m}\right)^2 = \Theta\left(\frac{k^2}{m}\right)$ chance of getting both edges of at least one of $T = \Theta(m)$ triangles, so need $k = \Omega(\sqrt{m})$.

Hard Case

One hard case is an *m*-star with triangles (possibly) completed by a matching.



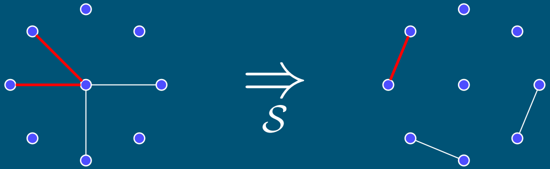
- Intuitively: if we keep k star edges, we have a $\sim T \times \left(\frac{k}{m}\right)^2 = \Theta\left(\frac{k^2}{m}\right)$ chance of getting both edges of at least one of $T = \Theta(m)$ triangles, so need $k = \Omega(\sqrt{m})$.
- Optimal up to log factors for general (classical) algorithms.

Sketching a Star Graph

In the two-player setting, our sketch solves this instance immediately in $O(\log n)$ space: make S the set of neighbors of the star vertex, and P the set of edges in the matching.

Sketching a Star Graph

In the two-player setting, our sketch solves this instance immediately in $O(\log n)$ space: make S the set of neighbors of the star vertex, and P the set of edges in the matching.



- Querying the sketch for # of pairs contained in S is equivalent to asking # of triangles in the graph.

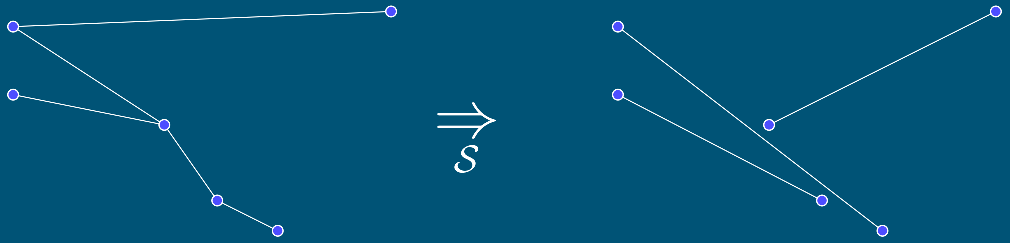
Sketching a Star Graph

In the two-player setting, our sketch solves this instance immediately in $O(\log n)$ space: make S the set of neighbors of the star vertex, and P the set of edges in the matching.

- Querying the sketch for # of pairs contained in S is equivalent to asking # of triangles in the graph.
- Implies exponential quantum algorithm for the two-player version of this *specific* instance of triangle counting.

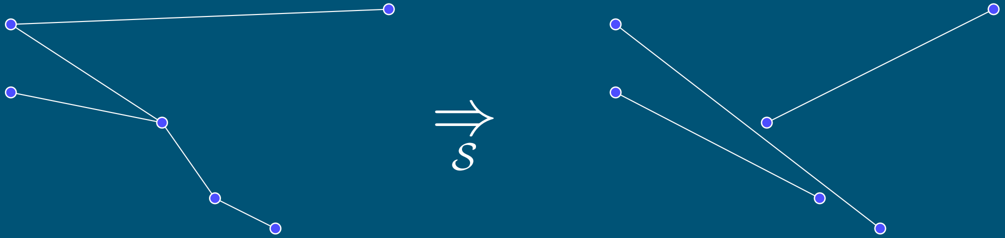
Sketching General Graphs

Generalizing to a general triangle-free graph (V, G) followed by a matching M is simple: make S the set of all edges in the first graph, and set $P = \{(uv, uw) : u \in V, vw \in M\}$.



Sketching General Graphs

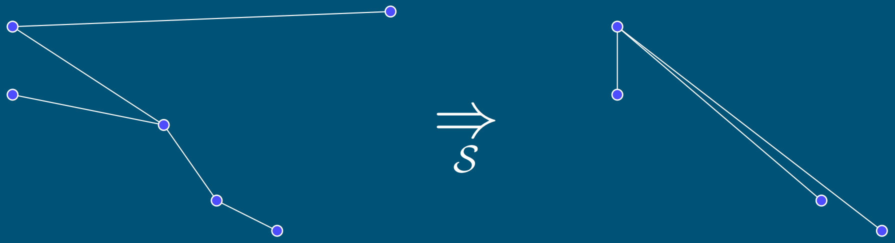
Generalizing to a general triangle-free graph (V, G) followed by a matching M is simple: make S the set of all edges in the first graph, and set $P = \{(uv, uw) : u \in V, vw \in M\}$.



Problems arise when the triangle-completing edges don't form a matching.

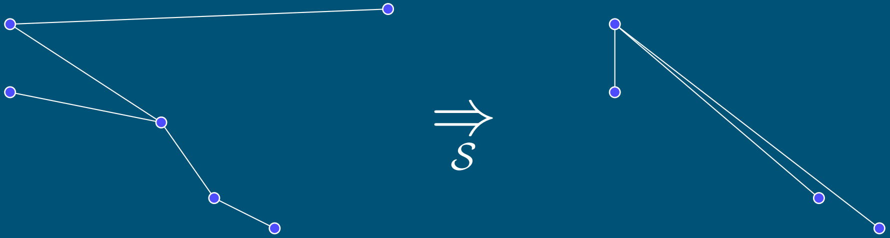
Sketching General Graphs

Our sketch doesn't work for non-matching edges because they don't give us disjoint pairs.



Sketching General Graphs

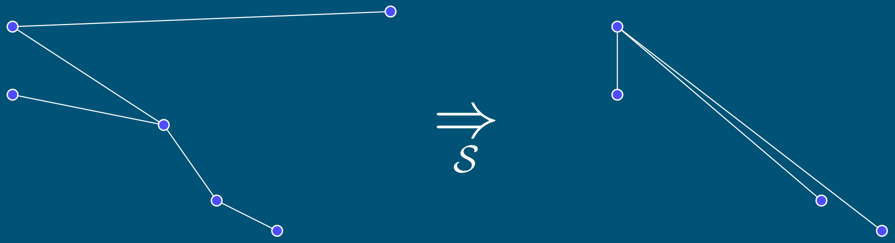
Our sketch doesn't work for non-matching edges because they don't give us disjoint pairs.



Requires copying sketch max-degree times, which is in general infeasible.

Sketching General Graphs

Our sketch doesn't work for non-matching edges because they don't give us disjoint pairs.

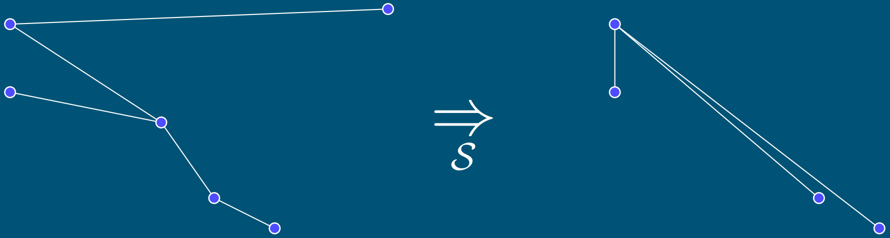


Requires copying sketch max-degree times, which is in general infeasible.

- Worst-case instances turn out to be "classically easy".

Sketching General Graphs

Our sketch doesn't work for non-matching edges because they don't give us disjoint pairs.



Requires copying sketch max-degree times, which is in general infeasible.

- Worst-case instances turn out to be "classically easy".
- Interpolation between classical and quantum estimators then allows a $O\left(n^{2/5}\right)$ v. $\Omega(\sqrt{n})$ quantum space advantage.

Sketching in the Stream

To make use of this in the stream, we need a couple more properties of the sketch:

Sketching in the Stream

To make use of this in the stream, we need a couple more properties of the sketch:

Swapping Given a pair (x, y), if $x \in S$ and $y \notin S$: remove x and add y. If $y \in S$ and $x \notin S$, do the opposite. Otherwise do nothing.

Sketching in the Stream

To make use of this in the stream, we need a couple more properties of the sketch:

Swapping Given a pair (x, y), if $x \in S$ and $y \notin S$: remove x and add y. If $y \in S$ and $x \notin S$, do the opposite. Otherwise do nothing.

Partial Query Given a pair p: return p and destroy S with probability $2/|S|$ if $p \subseteq S$. If p is not returned, set $S = S \setminus p$.

Sketching in the Stream

To make use of this in the stream, we need a couple more properties of the sketch:

Swapping Given a pair (x, y), if $x \in S$ and $y \notin S$: remove x and add y. If $y \in S$ and $x \notin S$, do the opposite. Otherwise do nothing.

Partial Query Given a pair p: return p and destroy S with probability $2/|S|$ if $p \subseteq S$. If p is not returned, set $S = S \setminus p$. (as with the full query setting, p can also be returned when $|p \cap S| = 1$, but will come with a random ± 1 sign)

Sketching in the Stream

To make use of this in the stream, we need a couple more properties of the sketch:

- **Swapping** Given a pair (x, y), if $x \in S$ and $y \notin S$: remove x and add y. If $y \in S$ and $x \notin S$, do the opposite. Otherwise do nothing.
- **Partial Query** Given a pair p: return p and destroy S with probability $2/|S|$ if $p \subseteq S$. If p is not returned, set $S = S \setminus p$. (as with the full query setting, p can also be returned when $|p \cap S| = 1$, but will come with a random ± 1 sign)

Excecuting a full query is then equivalent to doing a partial query for each $p \in P$ (in any order).

Quantum Advantage for Triangle Counting

- By starting with $S = [m]$ a set of m, "dummy variables", we can construct the sketch in the stream, swapping out one dummy variable for an edge whenever we see one.

Quantum Advantage for Triangle Counting

- By starting with $S = [m]$ a set of m, "dummy variables", we can construct the sketch in the stream, swapping out one dummy variable for an edge whenever we see one.
- By using partial queries, we can check whether an edge completes triangles with a pair of previously-arrived edges, without having to store that edge for later.

Quantum Advantage for Triangle Counting

- By starting with $S = [m]$ a set of m, "dummy variables", we can construct the sketch in the stream, swapping out one dummy variable for an edge whenever we see one.
- By using partial queries, we can check whether an edge completes triangles with a pair of previously-arrived edges, without having to store that edge for later.

This gives us quantum space advantage for counting triangles in the stream.

Theorem (Informal, K. '21)

There is a $\widetilde{O}\left(m^{2/5}\right)$-qubit streaming algorithm for counting triangles in the stream. ($\Omega(\sqrt{m})$ classically)

Interlude: Implementing the Streaming Properties of the Sketch

Swapping

To implement our swap operation, we need one more quantum primitive.

Unitary Evolution

A superposition $\sum_{x \in S} \alpha_x |x\rangle$ can be converted into a new one by any unitary (length-preserving) linear transformation.

Swapping

To implement our swap operation, we need one more quantum primitive.

Unitary Evolution

A superposition $\sum_{x \in S} \alpha_x |x\rangle$ can be converted into a new one by any unitary (length-preserving) linear transformation.

- One such transformation is swapping the basis elements $|x\rangle$ and $|y\rangle$.

Swapping

To implement our swap operation, we need one more quantum primitive.

Unitary Evolution

A superposition $\sum_{x \in S} \alpha_x |x\rangle$ can be converted into a new one by any unitary (length-preserving) linear transformation.

- One such transformation is swapping the basis elements $|x\rangle$ and $|y\rangle$.
- As our sketch is $\frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$, this will do nothing if neither or both of x and y are in S, and swap them if exactly one is.

Partial Queries

We can implement our partial query operation using the measurement postulate we already have.

Projective Measurement

Using labeled projectors (Λ_i) s.t. $\sum_i \Lambda_i = I$, measure state ψ. With probability $\|\Lambda_i \psi\|_2^2$, get result i and transform ψ to $\Lambda_i \psi / \|\Lambda_i \psi\|_2$.

Partial Queries

We can implement our partial query operation using the measurement postulate we already have.

Projective Measurement

Using labeled projectors (Λ_i) s.t. $\sum_i \Lambda_i = I$, measure state ψ. With probability $\|\Lambda_i \psi\|_2^2$, get result i and transform ψ to $\Lambda_i \psi / \|\Lambda_i \psi\|_2$.

- Rather than measure with a large collection of projectors, we use $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$ and the projector onto the rest of the space.

Partial Queries

We can implement our partial query operation using the measurement postulate we already have.

Projective Measurement

Using labeled projectors (Λ_i) s.t. $\sum_i \Lambda_i = I$, measure state ψ. With probability $\|\Lambda_i \psi\|_2^2$, get result i and transform ψ to $\Lambda_i \psi / \|\Lambda_i \psi\|_2$.

- Rather than measure with a large collection of projectors, we use $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$ and the projector onto the rest of the space.
- Our probabilities for returning $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$ are the same as they would have been when measuring it as a one of a large collection of pairs.

Partial Queries

We can implement our partial query operation using the measurement postulate we already have.

Projective Measurement

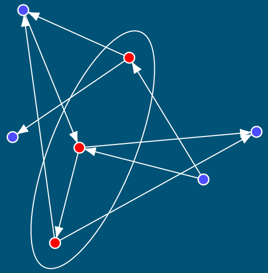
Using labeled projectors (Λ_i) s.t. $\sum_i \Lambda_i = I$, measure state ψ. With probability $\|\Lambda_i\psi\|_2^2$, get result i and transform ψ to $\Lambda_i\psi/\|\Lambda_i\psi\|_2$.

- Rather than measure with a large collection of projectors, we use $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$ and the projector onto the rest of the space.
- Our probabilities for returning $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$ are the same as they would have been when measuring it as a one of a large collection of pairs.
- If we do *not* return one of them, the superposition is projected onto the space orthogonal to $\mathrm{span}(|x\rangle, |y\rangle)$, i.e. x and y are deleted from S.

Application: Exponential Advantage for Maximum Directed Cut

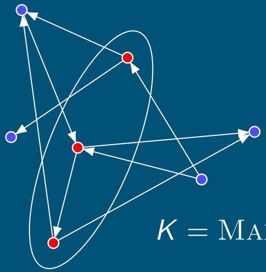
Maximum Directed Cut

Given a *directed* graph, what is the maximum, over partitions $V_0 \sqcap V_1 = V$, number of edges $\overrightarrow{uv}$ such that $u \in V_0$ and $v \in V_1$?



Maximum Directed Cut

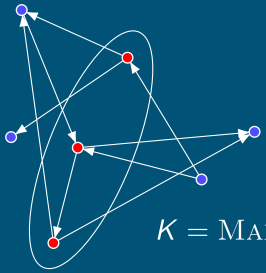
Given a *directed* graph, what is the maximum, over partitions $V_0 \sqcup V_1 = V$, number of edges $\overrightarrow{uv}$ such that $u \in V_0$ and $v \in V_1$?



$$K = \textsc{Max-DiCut}(G) = 5$$

Maximum Directed Cut

Given a *directed* graph, what is the maximum, over partitions $V_0 \sqcap V_1 = V$, number of edges $\overrightarrow{uv}$ such that $u \in V_0$ and $v \in V_1$?



$K = \textsc{Max-DiCut}(G) = 5$

We want to output an α-approximation in the minimum amount of space, i.e. $K' \in [\alpha K, K]$, using as little space as possible.

Complexity of Approximating Max-DiCut

- [Chou, Golovnev, Velusamy ‘20] Beating a $4/9$-approximation requires $\Omega(\sqrt{n})$ space classically.

Complexity of Approximating Max-DiCut

- [Chou, Golovnev, Velusamy '20] Beating a $4/9$-approximation requires $\Omega(\sqrt{n})$ space classically.
- [Feige, Jozeph '15] A $0.4835 > 4/9$-approximation is possible given a histogram of the number of edges going between vertices with *biases* in various ranges.

Complexity of Approximating Max-DiCut

- [Chou, Golovnev, Velusamy '20] Beating a $4/9$-approximation requires $\Omega(\sqrt{n})$ space classically.
- [Feige, Jozeph '15] A $0.4835 > 4/9$-approximation is possible given a histogram of the number of edges going between vertices with *biases* in various ranges.
- [Saxena, Singer, Sudan, Velusamy '23] This can be achieved in $\widetilde{O}(\sqrt{n})$ space classically.

Complexity of Approximating Max-DiCut

- [Chou, Golovnev, Velusamy '20] Beating a $4/9$-approximation requires $\Omega(\sqrt{n})$ space classically.
- [Feige, Jozeph '15] A $0.4835 > 4/9$-approximation is possible given a histogram of the number of edges going between vertices with *biases* in various ranges.
- [Saxena, Singer, Sudan, Velusamy '23] This can be achieved in $\widetilde{\mathrm{O}}(\sqrt{n})$ space classically.
- [**K.**, Parekh, Voronova] This can be achieved in polylog(n) space with our quantum sketch.

(First Order) Bias Histogram

The *bias* of a vertex v is $\frac{d_v^{\text{out}} - d_v^{\text{in}}}{d_v}$.

(First Order) Bias Histogram

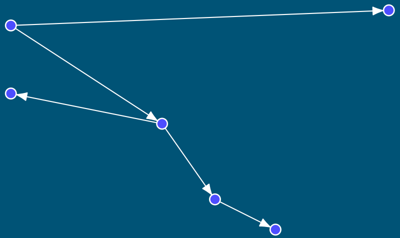
The *bias* of a vertex v is $\frac{d_v^{\text{out}} - d_v^{\text{in}}}{d_v}$.

		Head Bias			
		$-1 \leq b < -0.5$	$-0.5 \leq b < 0$	$0 \leq b < 0.5$	$0.5 \leq b \leq 1$
Tail Bias	$-0.5 \leq b < 0$	1245	2333	2974	9309
	$0 \leq b < 0.5$	9361	8421	82	66
	$0 \leq b < 0.5$	955	2133	5369	621
	$0.5 \leq b \leq 1$	3530	5312	4789	8472

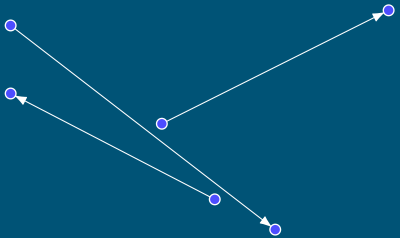
Given a partition of (B_i) of $[-1, 1]$ by thresholds, we want to know how many edges there are from B_i to B_j for each i, j.

Estimating the Bias Histogram

Return to the two-player setting: Alice and Bob both have graphs, and Alice wants to send Bob a message that will let him estimate their joint bias histogram.

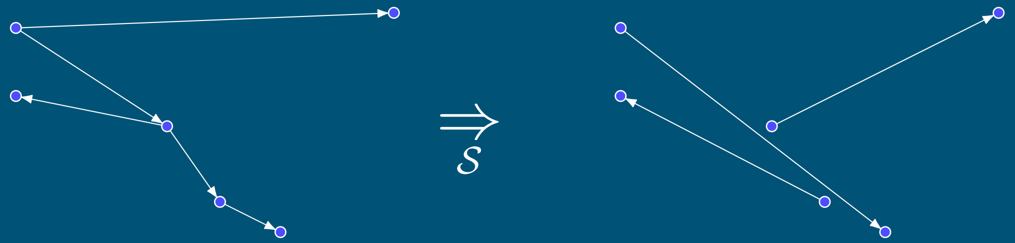


$\underset{\mathcal{S}}{\Rrightarrow}$



Estimating the Bias Histogram

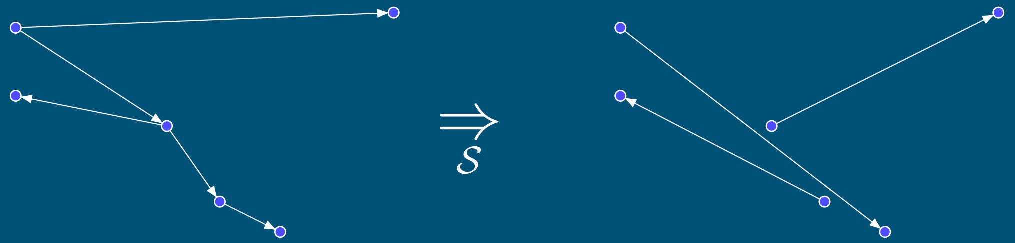
Return to the two-player setting: Alice and Bob both have graphs, and Alice wants to send Bob a message that will let him estimate their joint bias histogram.



- Alice's edges are easy: she can sample edges and send them to Bob with their endpoint out- and in- degrees.

Estimating the Bias Histogram

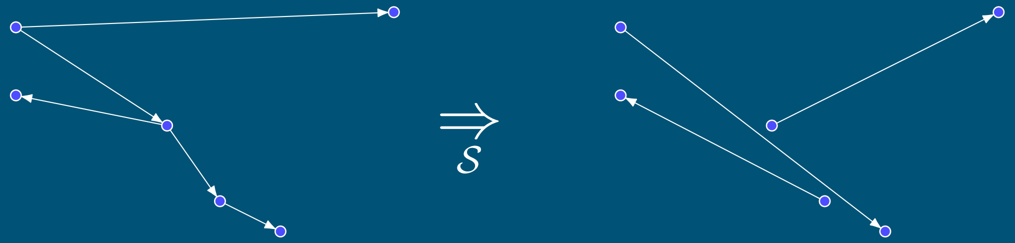
Return to the two-player setting: Alice and Bob both have graphs, and Alice wants to send Bob a message that will let him estimate their joint bias histogram.



- Alice's edges are easy: she can sample edges and send them to Bob with their endpoint out- and in- degrees.
- So the challenge is sampling head-tail degree pairs from Alice's graph that correspond to edges in Bob's graph.

Sketching for the Bias Histogram

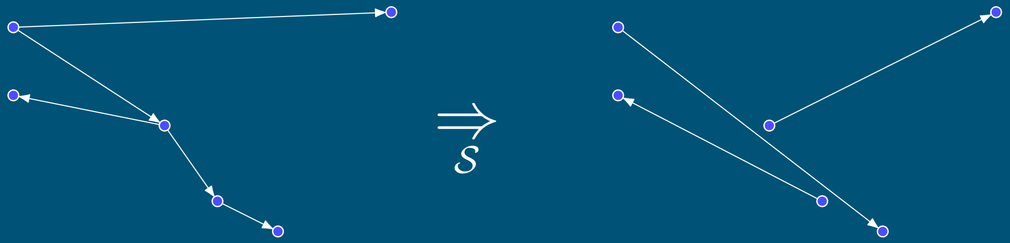
We can use our sketch to sample labeled pairs of Alice's vertices corresponding to Bob's edges, allowing him to calculate the biases of the endpoints of a sample of his edges with $O(\log(n))$ qubits of communication from Alice.



$$\underset{\mathcal{S}}{\Rrightarrow}$$

Sketching for the Bias Histogram

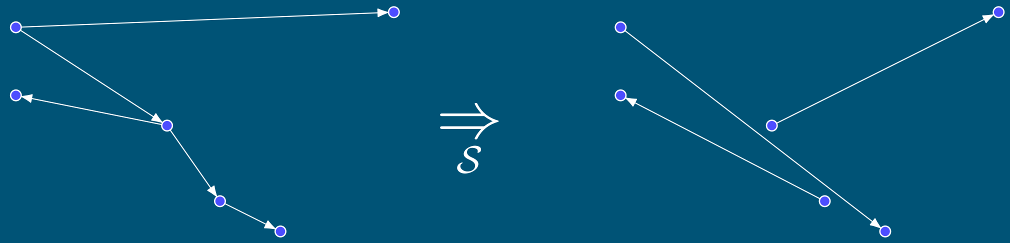
We can use our sketch to sample labeled pairs of Alice's vertices corresponding to Bob's edges, allowing him to calculate the biases of the endpoints of a sample of his edges with $O(\log(n))$ qubits of communication from Alice.



- Actually implementing this requires "coarsening" Alice's possible biases and having Bob query every possible coarsened bias pair for an edge.

Sketching for the Bias Histogram

We can use our sketch to sample labeled pairs of Alice's vertices corresponding to Bob's edges, allowing him to calculate the biases of the endpoints of a sample of his edges with $O(\log(n))$ qubits of communication from Alice.

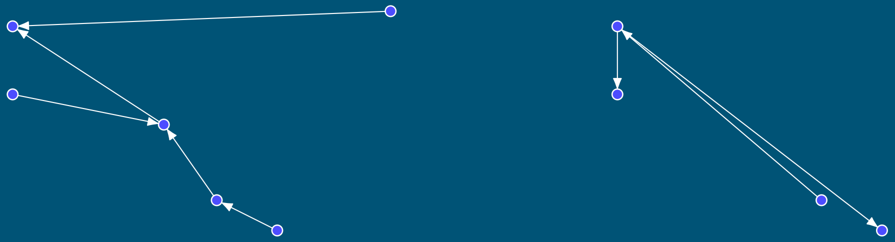


- Actually implementing this requires "coarsening" Alice's possible biases and having Bob query every possible coarsened bias pair for an edge.

Works if he has a matching. But what if he doesn't?

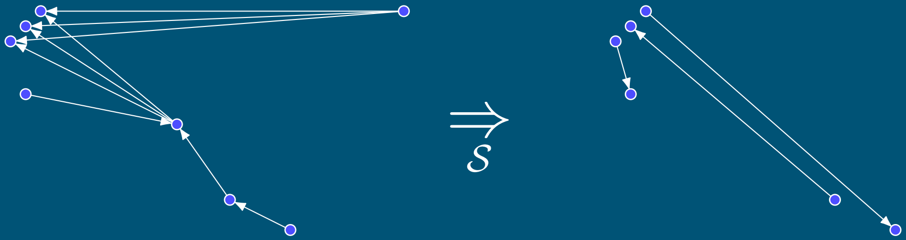
Copying

Suppose Alice knew how many edges Bob had incident to each of her vertices.



Copying

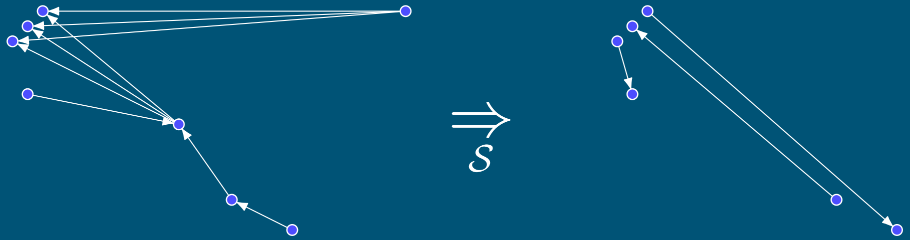
Suppose Alice knew how many edges Bob had incident to each of her vertices.



Now she can copy each vertex with corresponding multiplicity before sketching, and Bob can then measure with a matching on the copied vertices.

Copying

Suppose Alice knew how many edges Bob had incident to each of her vertices.



Now she can copy each vertex with corresponding multiplicity before sketching, and Bob can then measure with a matching on the copied vertices.

- She will still put at most $m = |E_{\text{Bob}}|$ elements total in her sketch set S, and so the needed property of $|p \in P : p \subseteq S| = \Omega(|S|)$ is preserved.

Guessing the Future

How can Alice copy correctly without knowing what Bob’s degrees will be?

Guessing the Future

How can Alice copy correctly without knowing what Bob's degrees will be?

- She can use her *own* degrees instead (multiplied by some large constant).

Guessing the Future

How can Alice copy correctly without knowing what Bob's degrees will be?

- She can use her *own* degrees instead (multiplied by some large constant).
- This works because when her degree for a vertex is much smaller than Bob's degree, that vertex's bias is almost exactly determined by Bob's input.

Guessing the Future

How can Alice copy correctly without knowing what Bob's degrees will be?

- She can use her *own* degrees instead (multiplied by some large constant).
- This works because when her degree for a vertex is much smaller than Bob's degree, that vertex's bias is almost exactly determined by Bob's input.
- We can therefore combine classical and quantum sampling to estimate the bias histogram in polylog(n) space.

Sketching in the Stream

Idealized algorithm:

Sketching in the Stream

Idealized algorithm:

- Maintain a sketch of $S = \{(v, b'_v) : v \in V\}$, where b'_v is some appropriate coarsening of the biases, and where (v, b'_v) is copied with multiplicity d_v.

Sketching in the Stream

Idealized algorithm:

- Maintain a sketch of $S = \{(v, b'_v) : v \in V\}$, where b'_v is some appropriate coarsening of the biases, and where (v, b'_v) is copied with multiplicity d_v.
- Query $((u, b'_u), (v, b'_v))$ for the possible biases (b'_u, b'_v) on seeing an edge $\overrightarrow{uv}$, then calculate biases for sampled vertices using the rest of the stream.

Sketching in the Stream

Idealized algorithm:

- Maintain a sketch of $S = \{(v, b'_v) : v \in V\}$, where b'_v is some appropriate coarsening of the biases, and where (v, b'_v) is copied with multiplicity d_v.
- Query $((u, b'_u), (v, b'_v))$ for the possible biases (b'_u, b'_v) on seeing an edge $\overrightarrow{uv}$, then calculate biases for sampled vertices using the rest of the stream.
- Use classical sampling to estimate edge counts for edges $\overrightarrow{uv}$ such that b_u or b_v are dominated by edges that arrive *after* $\overrightarrow{uv}$.

Sketching in the Stream

Problem: maintaining the S described would require knowing d_u, d_v when we see an edge $\overrightarrow{uv}$.

Sketching in the Stream

Problem: maintaining the S described would require knowing d_u, d_v when we see an edge $\overrightarrow{uv}$.

- Basic idea: we can copy a vertex every time an edge is seen incident to it, and use the number of times it has been copied to encode its degree.

Sketching in the Stream

Problem: maintaining the S described would require knowing d_u, d_v when we see an edge $\overrightarrow{uv}$.

- Basic idea: we can copy a vertex every time an edge is seen incident to it, and use the number of times it has been copied to encode its degree.
- On seeing an edge incident to v, add $(v, 0)$ to S after performing the swap $(v, i), (v, i + 1)$ for every i.

Sketching in the Stream

Problem: maintaining the S described would require knowing d_u, d_v when we see an edge $\overrightarrow{uv}$.

- Basic idea: we can copy a vertex every time an edge is seen incident to it, and use the number of times it has been copied to encode its degree.
- On seeing an edge incident to v, add $(v, 0)$ to S after performing the swap $(v, i), (v, i + 1)$ for every i.
- Now S contains $(v, 0) \ldots (v, d_v)$, and we can then query $((u, d_1), (v, d_2))$ if we want to sample restricted to edges with endpoint degrees at least (d_1, d_2).

Sketching in the Stream

Problem: maintaining the S described would require knowing d_u, d_v when we see an edge $\overrightarrow{uv}$.

- Basic idea: we can copy a vertex every time an edge is seen incident to it, and use the number of times it has been copied to encode its degree.
- On seeing an edge incident to v, add $(v, 0)$ to S after performing the swap $(v, i), (v, i + 1)$ for every i.
- Now S contains $(v, 0) \ldots (v, d_v)$, and we can then query $((u, d_1), (v, d_2))$ if we want to sample restricted to edges with endpoint degrees at least (d_1, d_2).
- Actual algorithm is complicated by need to track in- and out-degrees separately.

Exponential Quantum Advantage for Maximum Directed Cut

Theorem (Informal, K., Parekh, Voronova)

There is a $\operatorname{polylog}(n)$ *space quantum streaming algorithm that* 0.4835*-approximates the* $\textsc{Max-DiCut}$ *value of a graph.*

Exponential Quantum Advantage for Maximum Directed Cut

Theorem (Informal, K., Parekh, Voronova)

There is a polylog(n) *space quantum streaming algorithm that* 0.4835*-approximates the* MAX-DICUT *value of a graph.*

Contrasts with the undirected problem, where no quantum advantage is possible for any approximation ratio [Kapralov, Krachun '19], [**K.**, Parekh '22].

Conclusion

There is a simple, $O(\log n)$ space quantum sketch that allows sampling from a set based on set of disjoint pairs unknown at the time of sketching.

Conclusion

There is a simple, $O(\log n)$ space quantum sketch that allows sampling from a set based on set of disjoint pairs unknown at the time of sketching.

- Can be used to obtain quantum streaming space advantages for triangle counting.

Conclusion

There is a simple, $O(\log n)$ space quantum sketch that allows sampling from a set based on set of disjoint pairs unknown at the time of sketching.

- Can be used to obtain quantum streaming space advantages for triangle counting.
- And *exponential* advantages for $\textsc{Max-DiCut}$.

Conclusion

There is a simple, $O(\log n)$ space quantum sketch that allows sampling from a set based on set of disjoint pairs unknown at the time of sketching.

- Can be used to obtain quantum streaming space advantages for triangle counting.
- And *exponential* advantages for $\textsc{Max-DiCut}$.

Open Questions

- What is the correct complexity for triangle counting?

Conclusion

There is a simple, $O(\log n)$ space quantum sketch that allows sampling from a set based on set of disjoint pairs unknown at the time of sketching.

- Can be used to obtain quantum streaming space advantages for triangle counting.
- And *exponential* advantages for $\textsc{Max-DiCut}$.

Open Questions

- What is the correct complexity for triangle counting?
- Can we characterize which CSPs admit quantum space advantage in the stream?