Solving Graph Problems via Sketching and Streaming

Andrew McGregor
University of Massachusetts
Streaming

- **Input** Observe stream of edge insertions/deletions.
- **Goal** Using small memory, compute properties of the graph.
- **Classic Stream Results** Estimate statistics of numerical streams, such as quantiles, frequency moments, heavy hitters...
- **Graph Streams** Growing body of work on problems with more structure: distances, cuts, eigenvalues, random walks, clustering, matchings, dense components, vertex covers, hitting sets...

**Survey:** SIGMOD Record 2014

**Class Notes:** CMPSCI 711, UMass
https://people.cs.umass.edu/~mcgregor/courses/CS711S18/
**Sketching**

- **Random linear projection** \( M : \mathbb{R}^n \rightarrow \mathbb{R}^k \) where \( k \ll n \) that preserves properties of any \( v \in \mathbb{R}^n \) with high probability.

\[
\begin{bmatrix}
M \\
\end{bmatrix}
\begin{bmatrix}
v
\end{bmatrix}
= 
\begin{bmatrix}
Mv
\end{bmatrix}
\rightarrow \text{answer}
\]

- **Many results** for numerical statistics and basic geometric properties... *extensive theory* with connections to hashing, compressed sensing, dimensionality reduction, metric embeddings... *widely applicable* since embarrassingly parallelizable and suitable for stream processing.

? **Question** What about analyzing massive graphs via sketches?
Summary

- **Preliminaries** $L_0$ sampling and densest subgraph.
  
  “You can always do uniform sampling; sometimes it suffices.”

- **Matching Story** Using sketches to compute exact matchings, approximate matchings, and planar matchings.
  
  “Sketches enable interesting types of non-uniform sampling that are useful for graph problems.”

- **Connectivity Story** Using sketches to analyze edge and node connectivity, build cut and spectral sparsifiers etc.
  
  “Homomorphic compression: sketch first, compute later.”

- **Other Stories** Four small-space results we didn’t have space for.
part 0: Preliminaries
part 1: Matchings
part 2: Connectivity
part 3: Other Stories
Uniform Edge Sampling via Sketches

- \(L_0\) sampling Can use sketches to uniformly sample an edge from the graph stream using \(O(\text{polylog}(n))\) space. 
  
  Jowhari, Saglam, Tardos [PODS 11], Kapralov et al. [FOCS 17]

- Easy if there’s only edge insertions but non-trivial with insertions and deletions. Can treat result as a blackbox but will be important that the result is via linear sketches.
Application to Densest Subgraph

• Given a graph $G$, the \textit{density} of a set of nodes $S$ is:

$$D_S = \frac{\text{\# of edges with both endpoints in } S}{\text{\# of nodes in } S}$$

• \textit{Previous Result} $2+\varepsilon$ approx of max density $D^*$ in $\tilde{O}(\varepsilon^{-2} n)$ space. 
  \textit{Bhattycharya et al. [STOC 15], Bahmani et al. [PVLDB 12]}

• \textit{Our Result} One pass $1+\varepsilon$ approximation using $\tilde{O}(\varepsilon^{-2} n)$ space:

  Use $L_0$ sampling to uniformly sample $\tilde{O}(\varepsilon^{-2} n)$ edges. Let $\tilde{D}_S$ be estimate of $D_S$ based on sampled edges. Return $\max_S \tilde{D}_S$. 
  \textit{McGregor, Tench, Vorotnikova, Vu [MFCS 15]}

• \textit{Analysis} For any set of $k$ nodes $S$, with probability $1 - n^{-2k}$,

$$\tilde{D}_S = D_S \pm \varepsilon D^*$$

Use union bound over $O(n^k)$ subsets of size $k$ for each $k$. 
see also Mitzenmacher et al. [KDD 15], Esfandiari et al. [SPAA 16]
More details about $L_0$ sampling

- **$L_0$ Sampling**: There’s a random $M: \mathbb{R}^N \rightarrow \mathbb{R}^{\text{polylog } N}$ such that for any $a \in \mathbb{R}^N$, we can find random non-zero entry of $a$ from $Ma$ whp.

- Entry in $i^{\text{th}}$ row of $M$ is $1$ w/p $2^{-i+1}$. Some entry of $Ma$ probably corresponds to single entry of $a$.

\[
\begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x \\
x + y + z \\
x + z \\
0 \\
y \\
y \\
z \\
0 \\
0
\end{pmatrix} =
\begin{pmatrix}
x + y + z \\
x + z \\
y \\
0
\end{pmatrix}
\]

- Too many
- Too many
- Just right
- Too few

**Detail**: Need some extra tricks to a) recognize when entry of $Ma$ corresponds to a single entry of $a$ and b) determine the index of this entry.
part 0: Preliminaries
part 1: Matchings
part 2: Connectivity
part 3: Other Stories
Sometimes Uniform Sampling Isn’t Enough...

What other types of sampling a) are useful for solving graph problems and b) can be supported on dynamic graph streams via sketches?

Need to uniformly sample $\Omega(kn)$ edges before we find a matching of size $2k$. 

$\frac{n}{k} \gg k$
Matchings Story

- **Exact small matchings:** If matching has size $\leq k$ can find it exactly in $\tilde{O}(k^2)$ samples. Gives optimal stream algorithm.
  
  Chitnis et al. [SODA 16], Bury et al. [Algorithmica 18]

- **Approximate matching:** Find $t$-approx matching in $\tilde{O}(n^2/t^3)$ samples. Gives optimal stream algorithm.
  
  Chitnis et al. [SODA 16], Assadi, Khanna, Li, Yaroslavtsev [SODA 16]
  Related: Konrad [ESA 15], Bury, Schwiegelshohn [ESA 15]

- **Planar matching:** Can $5+\varepsilon$ approx matching size in planar graphs using $\tilde{O}(n^{4/5})$ space. Polylog space suffices if there are no edge deletions.
  
  Chitnis et al. [SODA 16], Bury, Schwiegelshohn [ESA 15]
  McGregor, Vorotnikova [APPROX 16], Cormode et al. [ESA 17]
  McGregor, Vorotnikova [SOSA 18]
**Matchings Story**

- **Exact small matchings**: If matching has size \( \leq k \) can find it exactly in \( \tilde{O}(k^2) \) samples. Gives optimal stream algorithm.
  
  *Chitnis et al. [SODA 16], Bury et al. [Algorithmica 18]*

- **Approximate matching**: Find \( t \)-approx matching in \( \tilde{O}(n^2/t^3) \) samples. Gives optimal stream algorithm.
  
  *Chitnis et al. [SODA 16], Assadi, Khanna, Li, Yaroslavtsev [SODA 16]*
  
  Related: *Konrad [ESA 15], Bury, Schwiegelshohn [ESA 15]*

- **Planar matching**: Can \( 5+\varepsilon \) approx matching size in planar graphs using \( \tilde{O}(n^{4/5}) \) space. Polylog space suffices if there are no edge deletions.

  *Chitnis et al. [SODA 16], Bury, Schwiegelshohn [ESA 15]*
  
  *McGregor, Vorotnikova [APPROX 16], Cormode et al. [ESA 17]*
  
  *McGregor, Vorotnikova [SOSA 18]*
Sample-Nodes-And-Pick-Edge Sampling

- To get a single SNAPE Sample:
  - Sample each node with probability $1/k$ and delete rest
  - Pick a random edge amongst those that remain.

**Theorem** If $G$ has max matching size $k$, then $O(k^2 \log k)$ SNAPE samples will include a max matching from $G$.

*Chitnis et al. [SODA 16], related: Bury, Schwiegelshohn [ESA 15]*
Consider a maximum matching \( M \) of size \( k \) and focus on arbitrary edge \( \{u,v\} \) in this matching.

- \( u \) and \( v \) only endpoints of \( M \) sampled with prob. \( \Omega(k^{-2}) \).
- Hence, when we pick one of the remaining edges it's either \( \{u,v\} \) or another edge that's equally useful.
- Take \( O(k^2 \log k) \) samples; apply analysis to all edges.
**Matchings Story**

- **Exact small matchings**: If matching has size $\leq k$ can find it exactly in $\tilde{O}(k^2)$ samples. Gives optimal stream algorithm.  
  *Chitnis et al. [SODA 16], Bury et al. [Algorithmica 18]*

- **Approximate matching**: Find $t$-approx matching in $O(n^2/t^3)$ samples. Gives optimal stream algorithm.  
  *Chitnis et al. [SODA 16], Assadi, Khanna, Li, Yaroslavtsev [SODA 16]*  
  related: *Konrad [ESA 15], Bury, Schwiegelshohn [ESA 15]*

- **Planar matching**: Can $5+\varepsilon$ approx matching size in planar graphs using $\tilde{O}(n^{4/5})$ space. Polylog space suffices if there are no edge deletions.  
  *Chitnis et al. [SODA 16], Bury, Schwiegelshohn [ESA 15]*  
  *McGregor, Vorotnikova [APPROX 16], Cormode et al. [ESA 17]*  
  *McGregor, Vorotnikova [SOSA 18]*
**Matchings Story**

- **Exact small matchings:** If matching has size $\leq k$ can find it exactly in $\tilde{O}(k^2)$ samples. Gives optimal stream algorithm.
  
  *Chitnis et al. [SODA 16], Bury et al. [Algorithmica 18]*

- **Approximate matching:** Find $t$-approx matching in $\tilde{O}(n^2/t^3)$ samples. Gives optimal stream algorithm.
  
  *Chitnis et al. [SODA 16], Assadi, Khanna, Li, Yaroslavtsev [SODA 16] related: Konrad [ESA 15], Bury, Schwiegelshohn [ESA 15]*

- **Planar matching:** Can $5+\varepsilon$ approx matching size in planar graphs using $\tilde{O}(n^{4/5})$ space. Polylog space suffices if there are no edge deletions.
  
A graph has **arboricity** \( a \) if any induced subgraph on \( r \) nodes has at most \( ar \) edges. For a planar graph \( a=3 \).

**Lemma:** \( \text{match}(G)/(2+a) \leq A \leq \text{match}(G) \) where \( A \) is total edge weight if each edge \( uv \) gets weight

\[
\chi_{uv} = \min \left( \frac{1}{\deg(u) + 1}, \frac{1}{\deg(v) + 1} \right)
\]

**Thm:** Can \( 2+a+\varepsilon \) approximate \( \text{match}(G) \) using \( \tilde{O}(n^{4/5}) \) space:

If \( \text{match}(G) \leq n^{2/5} \), can find exactly using earlier algorithm.

Otherwise, evaluate \( A \) on random set of \( \approx n^{4/5} \) nodes.

**Corollary:** \( 5+\varepsilon \) approx for planar graphs.
Proof of Lemma

- The edge weights are a **fractional matching**, i.e., for any node $u$:

  \[
  \sum_{v \in \Gamma(u)} x_{uv} \leq \sum_{v \in \Gamma(u)} \frac{1}{\deg(u) + 1} < 1
  \]

- To prove total weight $\leq \text{match}(G)$: Use Edmond's matching polytope thm since weight on subgraph of $r$ nodes is $\leq (r-1)/2$.

- To prove total weight $\geq \text{match}(G)/(2+\alpha)$:

  Total weight of edges incident to “high degree” vertices $H$ at least $|H|/(2+\alpha)$ and all other weights are at least $1/(2+\alpha)$.

  Matching size is at most $|H| + “edges not incident to H”$
part 0: Preliminaries
part 1: Matchings
part 2: Connectivity
part 3: Other Stories
Problem: n people each with a list of their friends amongst the group. In parallel, each sends a small number of bits to a central player who must determine if underlying graph is connected.

Thm: $O(\log^3 n)$ bits from each player suffices.

Any approach just using sampling fails... e.g., players can’t distinguish bridge edges from other edges in the graph.
Computing with sketches...

• **Problem**: n people each with a list of their friends amongst the group. In parallel, each sends a small number of bits to a central player who must determine if underlying graph is connected.

• **Thm**: $O(\log^3 n)$ bits from each player suffices.

• Any approach just using sampling fails... e.g., players can’t distinguish bridge edges from other edges in the graph.
Players send carefully-designed sketches of address books.

**Homomorphic Compression:** Instead of running algorithm on original data, run algorithm on sketched data.
Ingredient 1: Basic Algorithm

Algorithm (Spanning Forest):
- For each node: pick incident edge
- For each connected component: pick incident edge
- Repeat until no edges between connected comp.

Lemma After $O(\log n)$ rounds selected edges include spanning forest.
For node $i$, let $a_i$ be vector indexed by node pairs. Non-zero entries: $a_{i[i,j]} = 1$ if $j > i$ and $a_{i[i,j]} = -1$ if $j < i$.

\[
a_1 = \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]
\[
a_2 = \begin{pmatrix}
-1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]
\[
a_1 + a_2 = \begin{pmatrix}
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

Lemma For any subset of nodes $S \subset V$, non-zero entries of $\sum_{j \in S} a_j$ are edges across cut $(S, V \setminus S)$

Player $j$ sends $M(a_j)$ where $M$ is "$L_0$ sampling" sketch.
Recipe: Sketch & Compute on Sketches

- **Player with Address Books:** Player $j$ sends $M_{aj}$
- **Central Player:** “Runs Algorithm in Sketch Space”
  - Use $M_{aj}$ to get incident edge on each node $j$
  - For $i=2$ to $\log n$:
    - To get incident edge on component $S \subset V$ use:
      \[
      \sum_{j \in S} M_{aj} = M(\sum_{j \in S} a_j) \rightarrow \text{non-zero elt. of } \sum_{j \in S} a_j = \text{edge across cut}
      \]

**Detail:** Actually each player sends $\log n$ independent sketches $M_1a_j, M_2a_j, \ldots$ and central player uses $M_ia_j$ when emulating $i^{th}$ iteration of the algorithm.
Connectivity Story

- **Connectivity**: Test k-edge connectivity with $\tilde{O}(k)$ bit sketches.
  
  Ahn, Guha, McGregor [SODA 12]

- **Cut sparsification**: Estimating size of every cut up to $(1+\varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.
  
  Ahn, Guha, McGregor [PODS 12], Goel, Kapralov, Post [ArXiv 12]

- **Spectral sparsification**: Estimating eigenvalues up to $(1+\varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.
  
  Kapralov, Lee, Musco, Musco, Sidford [FOCS 14]
  related: Ahn, Guha, McGregor [APPROX 13], Kapralov, Woodruff [PODC 14]

- **Node connectivity**: Test k-node connectivity using $\tilde{O}(k^2)$ bits.
  
  Guha, McGregor, Tench [PODS 15]
Connectivity Story

• **Connectivity:** Test k-edge connectivity with $\tilde{O}(k)$ bit sketches.  
  Ahn, Guha, McGregor [SODA 12]

• **Cut sparsification:** Estimating size of every cut up to $(1+\varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.  
  Ahn, Guha, McGregor [PODS 12], Goel, Kapralov, Post [ArXiv 12]

• **Spectral sparsification:** Estimating eigenvalues up to $(1+\varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.  
  Kapralov, Lee, Musco, Musco, Sidford [FOCS 14]  
  related: Ahn, Guha, McGregor [APPROX 13], Kapralov, Woodruff [PODC 14]

• **Node connectivity:** Test k-node connectivity using $\tilde{O}(k^2)$ bits.  
  Guha, McGregor, Tench [PODS 15]
Extending to k-Edge-Connectivity

Algorithm: For i=1 to k:

- Let $F_i$ be spanning forest of $G(V,E-F_1-...-F_{i-1})$

Lemma: $F_1+...+F_k$ is k-edge-connected iff $G$ is.

Sketch: Simultaneously construct k independent connectivity sketches $M_1(G)$, $M_2(G)$, ..., $M_k(G)$.

Run Algorithm in Sketch Space:

- Use $M_1(G)$ to find a spanning forest $F_1$ of $G$
- Use $M_2(G)-M_2(F_1)=M_2(G-F_1)$ to find $F_2$
- Use $M_3(G)-M_3(F_1)-M_3(F_2)=M_3(G-F_1-F_2)$ to find $F_3$...

Extension: Can recover a set of “weak” edges whose removal leaves connected components with min-cut > k.
Connectivity Story

- **Connectivity:** Test k-edge connectivity with $\tilde{O}(k)$ bit sketches.  
  Ahn, Guha, McGregor [SODA 12]

- **Cut sparsification:** Estimating size of every cut up to $(1+\epsilon)$ factor with $\tilde{O}(\epsilon^{-2})$ bit sketches.  
  Ahn, Guha, McGregor [PODS 12], Goel, Kapralov, Post [ArXiv 12]

- **Spectral sparsification:** Estimating eigenvalues up to $(1+\epsilon)$ factor with $\tilde{O}(\epsilon^{-2})$ bit sketches.  
  Kapralov, Lee, Musco, Musco, Sidford [FOCS 14]  
  related: Ahn, Guha, McGregor [APPROX 13], Kapralov, Woodruff [PODC 14]

- **Node connectivity:** Test k-node connectivity using $\tilde{O}(k^2)$ bits.  
  Guha, McGregor, Tench [PODS 15]
Algorithm:

Let $G_0, G_1, \ldots, G_{O(\log n)}$ where $G_0 = G$, and $G$ formed from $G_{i-1}$ by deleting each edge with probability $1/2$.

Let $W_i$ be the set of edges in $G_i$ whose removal leaves a graph where non-zero cuts have $\Omega(\varepsilon^{-2} \log n)$ edges.

Lemma: Whp can estimate all cuts in $G_i$ given $W_i$ and $G_{i-1}$... can estimate all cuts in $G$ given $W_0, W_1, \ldots, W_{O(\log n)}$.

Sketch: For each $i$, use the $k$-edge-connectivity sketches of $G_i$ to find $W_i$. 
Connectivity Story

- **Connectivity:** Test k-edge connectivity with $\tilde{O}(k)$ bit sketches.
  
  Ahn, Guha, McGregor [SODA 12]

- **Cut sparsification:** Estimating size of every cut up to $(1 + \varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.
  
  Ahn, Guha, McGregor [PODS 12], Goel, Kapralov, Post [ArXiv 12]

- **Spectral sparsification:** Estimating eigenvalues up to $(1 + \varepsilon)$ factor with $\tilde{O}(\varepsilon^{-2})$ bit sketches.
  
  Kapralov, Lee, Musco, Musco, Sidford [FOCS 14]
  related: Ahn, Guha, McGregor [APPROX 13], Kapralov, Woodruff [PODC 14]

- **Node connectivity:** Test k-node connectivity using $\tilde{O}(k^2)$ bits.
  
  Guha, McGregor, Tench [PODS 15]
**Algorithm:**

Let $H_1, H_2, \ldots, H_r$ where $r = k^3 \log n$ and $H_i$ is induced subgraph on random set of $n/k$ nodes.

Let $F_i$ be a spanning forest of $H_i$.

**Lemma:** Whp, $F_1 + F_2 + \ldots + F_r$ is $k$-node connected iff $G$ is.

**Sketch:** Construct connectivity sketch for each $H_i$, and use this to find $F_i$. 

**k-Node-Connectivity**
part 0: Preliminaries
part 1: Matchings
part 2: Connectivity
part 3: Other Stories
Other Problems

- **Other Problems** Max-cut, clustering coefficients and triangles, degree distribution, correlation clustering, independent sets
  - Cormode et al. [ICML 15], Bulteau et al. [Algorithmic 16], McGregor et al. [PODS 16], Cormode, Dark, Konrad [arXiv 17], Kapralov et al. [SODA 17]

- If stream consists of m subsets of nodes...

- **Set-cover** $\Theta(mn/t)$ space for t-approx in one pass also tight space/pass/approx. trade-offs.
  - Assadi et al. [STOC 16]
  - Chakrabarti, Wirth [SODA 16], Har-Peled et al. [PODS 16]

- **Hitting set** Optimal hitting of size k in $\tilde{O}(k^d)$ if all sets have size at most d.
  - Chitnis et al. [SODA 16]

- **Max k-Coverage** $1-\varepsilon$ approx in $\tilde{O}(m\varepsilon^{-2}\min(\varepsilon^{-1},k))$ space or 0.5 approx in $\tilde{O}(k)$ space.
  - McGregor, Vu [ICDT 17], Bateni et al. [SPAA 17], Assadi [PODS 17]
Clustering + Maximal Independent Set

- Consider a complete graph where edges are labelled attractive or repulsive. Given a node partition, an attractive edge is sad if it is cut and a repulsive edge is sad if it is not cut.

- **Correlation Clustering** Find partition minimizing # sad edges.
  
  See tutorial Bonchi, Garcia-Soriano, Liberty [KDD 14]

- **3-Approx Algorithm**
  
  a) Pick random node.  
  
  b) Form cluster with it and its attracted neighbors.  
  
  c) Remove cluster from graph and repeat until nodes remain.  
  
  Ailon, Charikar, Newman [J.ACM 08]
Clustering + Maximal Independent Set

- **Emulating algorithm in two passes:**
  - **Preprocess** Randomly order nodes, $v_1, v_2, \ldots, v_n$.
  - **First Pass** Store all attractive edges incident to $\{v_1, \ldots, v_{\sqrt{n}}\}$. Now can emulate first $\sqrt{n}$ iterations of the algorithm.
  - **Second Pass** Store all remaining attractive edges. Now can emulate remaining steps of the algorithm.

- **Thm** Algorithm uses $\tilde{O}(n^{1.5})$ space.  
  *Ahn et al. [ICML 16]*

- **Proof Idea** At most $n^{1.5}$ edges stored in first pass. In second pass, can show remaining node have at most $n^{0.5}$ neighbors.

- With more work, can get $\tilde{O}(n)$ space with $O(\log \log n)$ passes. Can also find maximal independent sets.
Coloring

- **Coloring** With min number of colors, assign a color to every node such that no edge has monochromatic endpoints.

- **Thm** Can color a graph in $\Delta+1$ colors where $\Delta$ is max degree.

  How can we do this in a few passes with $\tilde{O}(n)$ space?

  - $O(\Delta \log \log n)$ passes via independent sets. Let’s do better!
**Coloring**

- \((1+\varepsilon)\Delta\) Coloring
  - a) Randomly color with \(\Delta/r\) colors.
  - b) Store edges \(E'\) with monochromatic endpoints.
  - c) Shade colors such that \(E'\) edges no longer monochromatic.

  \[\text{Bera, Ghosh [ArXiv 18]}\]

- **Space Analysis**
  \(|E'|=O(nr)\) since probability edge in \(E'\) is \(r/\Delta\).

- **Colors Analysis**
  If \(r \approx \varepsilon^{-2} \log n\), max degree in \(E'\) is \(\Delta_{E'}<(1+\varepsilon)r\) and final number of colors is \((1+\Delta_{E'})\Delta/r = (1+\varepsilon)\Delta\).

- \(\Delta+1\) Coloring Idea
  For node \(v\), pick \(S_v \subset \{1, \ldots, \Delta+1\}\) of \(O(\log n)\) random colors. May assume \(v\)'s color in \(S_v\).

  \[\text{Assadi et al. [SODA 19]}\]
Wrapping Up

- **Matching Story** Using sketches to compute exact matchings, approximate matchings, and planar matchings.
  
  “Sketches enable interesting types of sampling via sketches that are useful for graph problems.”

- **Connectivity Story** Using sketches to analyze edge and node connectivity, build cut and spectral sparsifiers etc.
  
  “Homomorphic compression: sketch first, compute later.”

- **Other Stories** Densest subgraph, clustering coefficients and triangles, correlation clustering, set cover and hitting sets…

Thanks!