Streaming Set Cover

Amit Chakrabarti

Dartmouth College

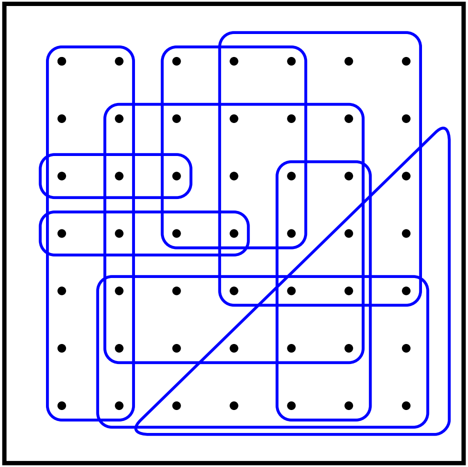
Joint work with A. Wirth

Sublinear Algorithms Workshop
JHU, Jan 2016

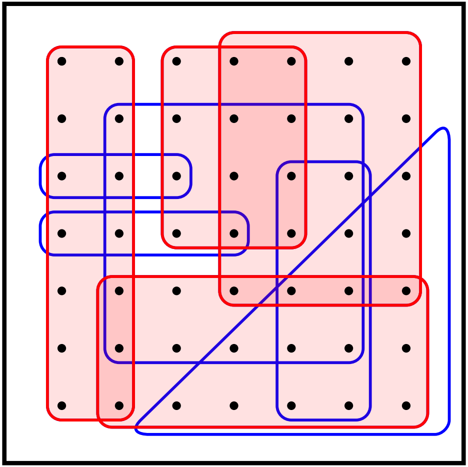
Combinatorial Optimisation Problems

- 1950s, 60s: Operations research
- 1970s, 80s: NP-hardness
- 1990s, 2000s: Approximation algorithms, hardness of approximation
- 2010s: Space-constrained settings, e.g., streaming

Set Cover



Set Cover



Set Cover with Sets Streamed

- Input: stream of m sets, each $\subseteq [n]$
- Goal: cover universe $[n]$ using as few sets as possible

Set Cover with Sets Streamed

- Input: stream of m sets, each $\subseteq [n]$
- Goal: cover universe $[n]$ using as few sets as possible
  - Use sublinear (in m) space
  - Ideally $O(n \operatorname{polylog} n)$ ... "semi-streaming"
  - Need $\Omega(n \log n)$ space to *certify*: for each item, who covered it?

Think $m \geq n$

Background and Related Work

Offline results:

- Best possible poly-time approx $(1 \pm o(1)) \ln n$ [Johnson'74] [Slavík'96] [Lund-Yannakakis'94] [Dinur-Steurer'14]
- Simple greedy strategy gets $\ln n$-approx:
  - Repeatedly add set with highest *contribution*
  - Contribution := number of *new* elements covered

Background and Related Work

Offline results:

- Best possible poly-time approx $(1 \pm o(1)) \ln n$ [Johnson'74] [Slavík'96] [Lund-Yannakakis'94] [Dinur-Steurer'14]
- Simple greedy strategy gets $\ln n$-approx:
  - Repeatedly add set with highest *contribution*
  - Contribution := number of *new* elements covered

Streaming results:

- One pass semi-streaming $O(\sqrt{n})$ approx
- This is best possible in one semi-streaming pass [Emek-Rosén'14]
- $O(\log n)$ semi-streaming passes allow $O(\log n)$ approx [Saha-Getoor'09] [Cormode-Karloff-Wirth'10]

Background and Related Work

Offline results:

- Best possible poly-time approx $(1 \pm o(1)) \ln n$ [Johnson'74] [Slavík'96] [Lund-Yannakakis'94] [Dinur-Steurer'14]
- Simple greedy strategy gets $\ln n$-approx:
  - Repeatedly add set with highest *contribution*
  - Contribution := number of *new* elements covered

Streaming results:

- One pass semi-streaming $O(\sqrt{n})$ approx
- This is best possible in one semi-streaming pass [Emek-Rosén'14]
- $O(\log n)$ semi-streaming passes allow $O(\log n)$ approx [Saha-Getoor'09] [Cormode-Karloff-Wirth'10]
- There's more: wait till the end! [Nisan'02] [Demaine-Indyk-Mahabadi-Vakilian'14] [Indyk-M-V'16]

Related Work: In Greater Detail

Algorithms using p passes, S space, giving α-approximation

Upper bounds:

- $p = 1, S = \widetilde{O}(n), \alpha = O(\sqrt{n})$ [Emek-Rosén'14]
- $p = O(\log n), S = \widetilde{O}(n), \alpha = O(\log n)$ [Cormode-Karloff-Wirth'10]
- $S = \widetilde{O}(mn^{1/\Omega(\log p)}), \alpha = O(p)$ [Demaine-Indyk-Mahabadi-Vakilian'14]
- $S = \widetilde{O}(mn^{1/\Omega(p)}), \alpha = O(p)$ [Indyk-Mahabadi-Vakilian'16]

Lower bounds:

- $p = 1, S = \widetilde{O}(n) \Rightarrow \alpha = \Omega(n^{1/2-\delta})$ [Emek-Rosén'14]
- $\alpha < \frac{1}{2}\log_2 n \Rightarrow S = \Omega(m)$ [Nisan'02]
- $\alpha = O(1)$, deterministic $\Rightarrow S = \Omega(mn)$ [Demaine-I-M-V'14]
- $\alpha = 1 \Rightarrow S = \widetilde{\Omega}(n^{1+1/(2(p+1))})$ [Indyk-Mahabadi-Vakilian'16]
- $p = 1, \alpha = \frac{3}{2} \Rightarrow S = \Omega(mn)$ [Indyk-Mahabadi-Vakilian'16]

Our Results

Upper bound

- With p passes, semi-streaming space, get $O(n^{1/(p+1)})$-approx
- Algorithm giving this approx based on very simple heuristic
- Deterministic

Lower bound

- Randomised
- In p passes, semi-streaming space, need $\Omega(n^{1/(p+1)}/p^2)$ approx
- Upper bound tight for all constant p
- Semi-streaming $O(\log n)$ approx requires $\Omega(\log n / \log \log n)$ passes

Progressive Greedy Algorithm

Recall simple greedy:

- Repeatedly add set with highest *contribution*
- Contribution := number of *new* elements covered

Progressive greedy:

- In first pass, add all sets with contribution $\geq n^{1-1/p}$
- In second pass, add all sets with contribution $\geq n^{1-2/p}$
- ...
- ...
- In pth pass, add all sets with contribution ≥ 1

Progressive Greedy Algorithm

procedure $\textsc{GreedyPass}$(stream σ, threshold τ, set Sol, array $\mathit{Coverer}$)
    for each set S_i in σ **do**
        $C \leftarrow \{x : \mathit{Coverer}[x] \neq 0\}$ ▷ the already covered elements
        if $|S_i \setminus C| \geq \tau$ **then** ▷ set's contribution $\geq$ threshold
            $\mathit{Sol} \leftarrow \mathit{Sol} \cup \{i\}$
            for each $x \in S_i \setminus C$ **do** $\mathit{Coverer}[x] \leftarrow i$

procedure $\textsc{ProgGreedyNaive}$(stream σ, integer n, integer $p \geq 1$)
    $\mathit{Coverer}[1 \ldots n] \leftarrow 0^n$; $\mathit{Sol} \leftarrow \varnothing$
    for $j = 1$ **to** p **do** $\textsc{GreedyPass}(\sigma, n^{1-j/p}, \mathit{Sol}, \mathit{Coverer})$
    output $\mathit{Sol}, \mathit{Coverer}$

Progressive Greedy: Analysis Idea

Consider $p = 2$ passes

- First pass: admit sets iff contribution $\geq \sqrt{n}$
- Thus, first pass adds at most $\sqrt{n}$ sets to Sol

Progressive Greedy: Analysis Idea

Consider $p = 2$ passes

- First pass: admit sets iff contribution $\geq \sqrt{n}$
- Thus, first pass adds at most $\sqrt{n}$ sets to Sol
- Second pass: Opt covers remaining items with sets of contrib $\leq \sqrt{n}$
- Thus, Sol will cover the same using $\leq \sqrt{n}|Opt|$ sets

Progressive Greedy: Analysis Idea

Consider $p = 2$ passes

- First pass: admit sets iff contribution $\geq \sqrt{n}$
- Thus, first pass adds at most $\sqrt{n}$ sets to Sol
- Second pass: Opt covers remaining items with sets of contrib $\leq \sqrt{n}$
- Thus, Sol will cover the same using $\leq \sqrt{n}|Opt|$ sets

But wait, this uses *two* passes for $O(\sqrt{n})$ approx!

Progressive Greedy: Analysis Idea

Consider $p = 2$ passes

- First pass: admit sets iff contribution $\geq \sqrt{n}$
- Thus, first pass adds at most $\sqrt{n}$ sets to Sol
- Second pass: Opt covers remaining items with sets of contrib $\leq \sqrt{n}$
- Thus, Sol will cover the same using $\leq \sqrt{n}|Opt|$ sets

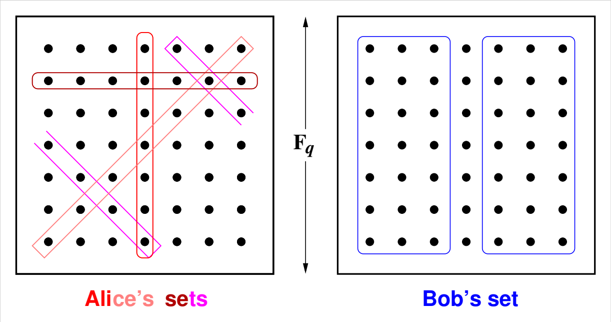
But wait, this uses *two* passes for $O(\sqrt{n})$ approx!

- Logic of last pass especially simple: add set if positive contrib
- Can *fold* this into previous one

Final result: p passes, $O(n^{1/(p+1)})$-approx

Lower Bound Idea: One Pass

Reduce from INDEX: Alice gets $x \in \{0,1\}^n$, Bob gets $j \in [n]$, Alice talks to Bob, who must determine x_j. Requires $\Omega(n)$-bit message. [Ablayev'96]

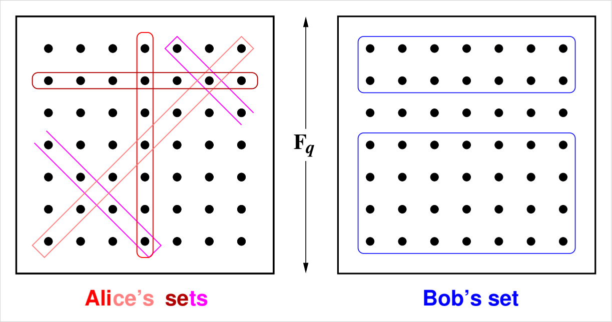


Universe $\mathbb{F}_q^2$

$n = q^2$

Lower Bound Idea: One Pass

Reduce from INDEX: Alice gets $x \in \{0,1\}^n$, Bob gets $j \in [n]$, Alice talks to Bob, who must determine x_j. Requires $\Omega(n)$-bit message. [Ablayev'96]



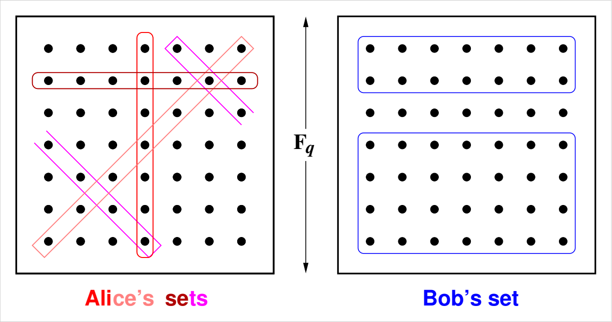
Universe $\mathbb{F}_q^2$

$n = q^2$

If Alice has Bob's *missing line*, then $|Opt| = 2$, else $|Opt| \geq q$

Lower Bound Idea: One Pass

Reduce from INDEX: Alice gets $x \in \{0,1\}^n$, Bob gets $j \in [n]$, Alice talks to Bob, who must determine x_j. Requires $\Omega(n)$-bit message. [Ablayev'96]



Universe $\mathbb{F}_q^2$

$n = q^2$

If Alice has Bob's *missing line*, then $|Opt| = 2$, else $|Opt| \geq q$

So $\Theta(\sqrt{n})$ approx requires $\Omega(\#\text{lines}) = \Omega(q^2) = \Omega(n)$ space

Next Steps

Goal: p semi-streaming passes require $\Omega(n^{1/(p+1)})$ approx

- Handle more passes
- Increase space bound

Next Steps

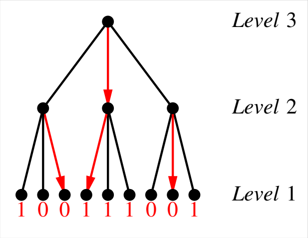
Goal: p semi-streaming passes require $\Omega(n^{1/(p+1)})$ approx

- Handle more passes
  - Can't start from INDEX, need harder communication problem
- Increase space bound
  - Need $\omega(n)$ to rule out semi-streaming

Tree Pointer Jumping

Multiplayer game $\text{TPJ}_{p+1,t}$ defined on complete $(p+1)$-level t-ary tree

- Pointer to child at each internal level-i node (known to Player i)
- Bit at each leaf node (known to Player 1)
- Goal: output (whp) bit reached by following pointers from root



Model: p rounds of communication

Each round:

$\text{PLAYER}_1, \text{PLAYER}_2, \ldots, \text{PLAYER}_{p+1}$

Theorem: Longest message is $\Omega(t/p^2)$ bits [C.-Cormode-McGregor'08]

Multi-Pass Set Cover: First Attempt

Two passes, reducing from $\text{TPJ}_{3,t}$, using universe $\mathbb{F}_q^3$ (so $n = q^3$)

- Three players: Alice, Bob, Carol
  - Alice encodes leaf bits: lines in $\mathbb{F}_q^3$
  - Bob encodes lower pointers: planes in $\mathbb{F}_q^3$ with a line deleted
  - Carol encodes root pointer: $\mathbb{F}_q^3$ with a plane deleted

Multi-Pass Set Cover: First Attempt

Two passes, reducing from $\mathrm{TPJ}_{3,t}$, using universe $\mathbb{F}_q^3$ (so $n = q^3$)

- Three players: Alice, Bob, Carol
  - Alice encodes leaf bits: lines in $\mathbb{F}_q^3$
  - Bob encodes lower pointers: planes in $\mathbb{F}_q^3$ with a line deleted
  - Carol encodes root pointer: $\mathbb{F}_q^3$ with a plane deleted
- (Carol set) $\cup$ (corresp. Bob set) $= \mathbb{F}_q^3 \setminus$ (a line)
- If Alice has the missing line, then $|Opt| = 3$, else $\Rightarrow |Opt| \geq q$ (*)

Multi-Pass Set Cover: First Attempt

Two passes, reducing from $\text{TPJ}_{3,t}$, using universe $\mathbb{F}_q^3$ (so $n = q^3$)

- Three players: Alice, Bob, Carol
  - Alice encodes leaf bits: lines in $\mathbb{F}_q^3$
  - Bob encodes lower pointers: planes in $\mathbb{F}_q^3$ with a line deleted
  - Carol encodes root pointer: $\mathbb{F}_q^3$ with a plane deleted
- (Carol set) $\cup$ (corresp. Bob set) $= \mathbb{F}_q^3 \setminus$ (a line)
- If Alice has the missing line, then $|Opt| = 3$, else $\Rightarrow |Opt| \geq q$ (*)

How good is this?

Multi-Pass Set Cover: First Attempt

Two passes, reducing from $\mathrm{TPJ}_{3,t}$, using universe $\mathbb{F}_q^3$ (so $n = q^3$)

- Three players: Alice, Bob, Carol
  - Alice encodes leaf bits: lines in $\mathbb{F}_q^3$
  - Bob encodes lower pointers: planes in $\mathbb{F}_q^3$ with a line deleted
  - Carol encodes root pointer: $\mathbb{F}_q^3$ with a plane deleted
- (Carol set) $\cup$ (corresp. Bob set) $= \mathbb{F}_q^3 \setminus$ (a line)
- If Alice has the missing line, then $|Opt| = 3$, else $\Rightarrow |Opt| \geq q$ (*)

How good is this?

- Each pointer encoded by Bob can choose from only as many leaves as there are lines in a specific plane $\implies t = \Theta(q^2) = \Theta(n^{2/3})$

Multi-Pass Set Cover: First Attempt

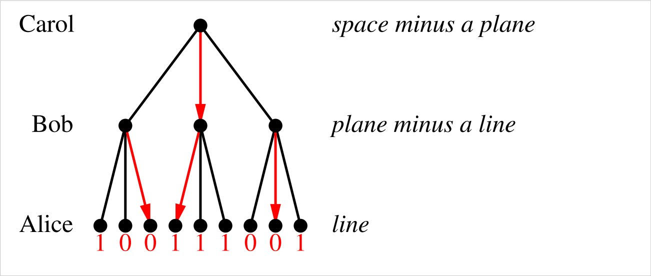
Two passes, reducing from $\text{TPJ}_{3,t}$, using universe $\mathbb{F}_q^3$ (so $n = q^3$)

- Three players: Alice, Bob, Carol
  - Alice encodes leaf bits: lines in $\mathbb{F}_q^3$
  - Bob encodes lower pointers: planes in $\mathbb{F}_q^3$ with a line deleted
  - Carol encodes root pointer: $\mathbb{F}_q^3$ with a plane deleted
- (Carol set) $\cup$ (corresp. Bob set) $= \mathbb{F}_q^3 \setminus$ (a line)
- If Alice has the missing line, then $|Opt| = 3$, else $\Rightarrow |Opt| \geq q$ (*)

How good is this?

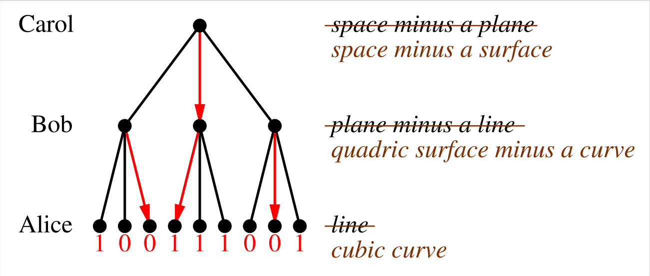
- Each pointer encoded by Bob can choose from only as many leaves as there are lines in a specific plane $\implies t = \Theta(q^2) = \Theta(n^{2/3})$
- Implies space $\Omega(n^{2/3})$ for approx $< q/3 = \Theta(n^{1/3})$

Insight



Too few lines in a plane...

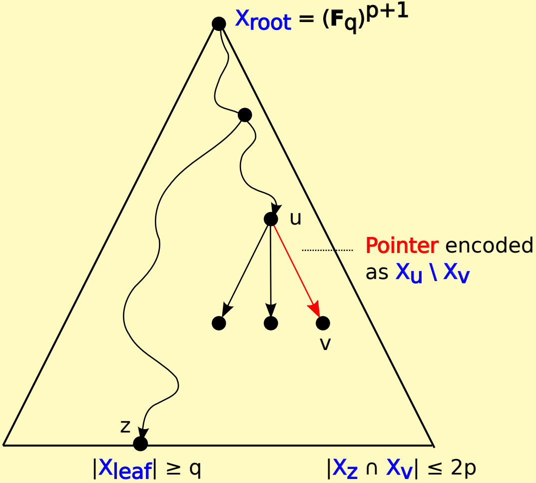
Insight



Too few lines in a plane... increase the degree!

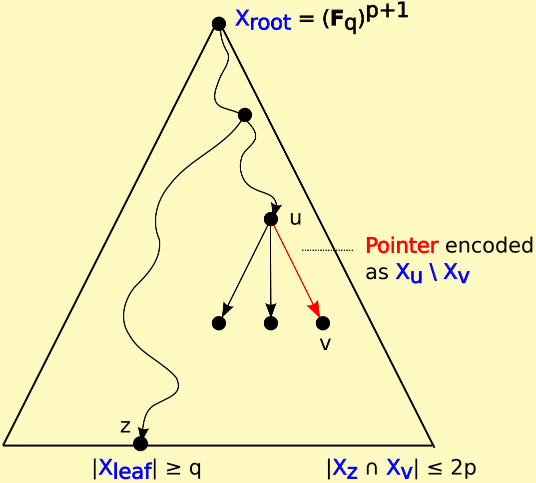
Edifices

Basic idea: Generalise affine plane to high-rank Buekenhout geometry



Edifices

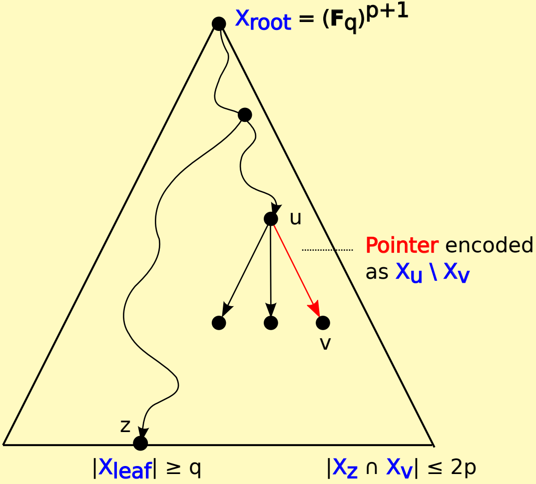
Basic idea: Generalise affine plane to high-rank Buekenhout geometry



- Universe $\mathbb{F}_q^{p+1}$
- Variety X_u at node u
- u above v
  $\implies X_u \supseteq X_v$

Edifices

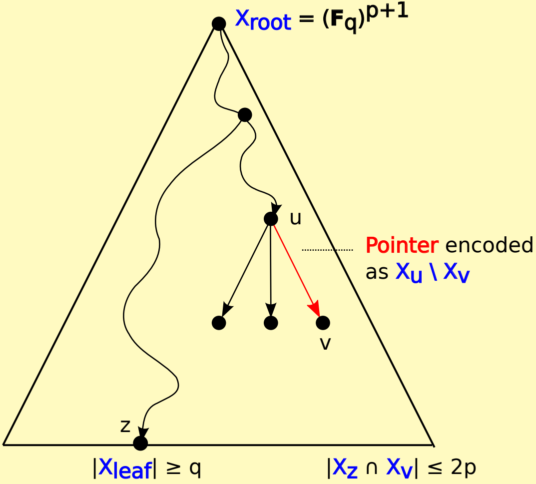
Basic idea: Generalise affine plane to high-rank Buekenhout geometry



- Universe $\mathbb{F}_q^{p+1}$
- Variety X_u at node u
- u above v
  $\implies X_u \supseteq X_v$
- Leaf z with bit $= 1$ encoded as set X_z

Edifices

Basic idea: Generalise affine plane to high-rank Buekenhout geometry



- Universe $\mathbb{F}_q^{p+1}$
- Variety X_u at node u
- u above v
  $\implies X_u \supseteq X_v$
- Leaf z with bit $= 1$ encoded as set X_z
- If player 1 has the *missing variety*, then $|Opt| = p+1$, else $|Opt| \geq q/(2p)$

Construction of an Edifice

Basic idea: Varieties at leaves are low-degree curves, at level 2 they are low-degree surfaces, and so on.

Concern: Determining “cardinality” of algebraic variety over finite field is the stuff of difficult mathematics.

Construction of an Edifice

Basic idea: Varieties at leaves are low-degree curves, at level 2 they are low-degree surfaces, and so on.

Concern: Determining "cardinality" of algebraic variety over finite field is the stuff of difficult mathematics.

Our Solution: Define varieties using equations of special format

- Coordinates $(x, y_1, y_2, \ldots, y_p)$
- Equation at each edge of tree; at level i:

$$y_i = a_1 y_1 + \cdots a_{i-1} y_{i-1} + a_i f_{p+1-i}(x)$$
$$f_j(x) = \text{monic poly in } \mathbb{F}_q[x] \text{ of degree } p + j$$

- Variety X_u defined by equations on root-to-u path

Construction of an Edifice

Basic idea: Varieties at leaves are low-degree curves, at level 2 they are low-degree surfaces, and so on.

Concern: Determining "cardinality" of algebraic variety over finite field is the stuff of difficult mathematics.

Our Solution: Define varieties using equations of special format

- Coordinates $(x, y_1, y_2, \ldots, y_p)$
- Equation at each edge of tree; at level i:

$$y_i = a_1 y_1 + \cdots a_{i-1} y_{i-1} + a_i f_{p+1-i}(x)$$
$$f_j(x) = \text{monic poly in } \mathbb{F}_q[x] \text{ of degree } p + j$$

Cardinality bound via much simpler mathematics.

- Schwartz-Zippel lemma
- Linear independence arguments via row reduction

Recap: Related Work and Our Results

Upper bounds (p passes, S space, α-approximation):

- $p = 1, S = \widetilde{O}(n), \alpha = O(\sqrt{n})$ [Emek-Rosén'14]
- $p = O(\log n), S = \widetilde{O}(n), \alpha = O(\log n)$ [Cormode-Karloff-Wirth'10]
- $S = \widetilde{O}(mn^{1/\Omega(\log p)}), \alpha = O(p)$ [Demaine-Indyk-Mahabadi-Vakilian'14]
- $S = \widetilde{O}(mn^{1/\Omega(p)}), \alpha = O(p)$ [Indyk-Mahabadi-Vakilian'16]
- $S = \widetilde{O}(n), \alpha = O(pn^{1/(p+1)})$ [this work]

Lower bounds:

- $p = 1, S = \widetilde{O}(n) \Rightarrow \alpha = \Omega(n^{1/2-\delta})$ [Emek-Rosén'14]
- $\alpha < \frac{1}{2}\log_2 n \Rightarrow S = \Omega(m)$ [Nisan'02]
- $\alpha = O(1)$, deterministic $\Rightarrow S = \Omega(mn)$ [Demaine-I-M-V'14]
- $\alpha = 1 \Rightarrow S = \widetilde{\Omega}(n^{1+1/(2(p+1))})$ [Indyk-Mahabadi-Vakilian'16]
- $p = 1, \alpha = \frac{3}{2} \Rightarrow S = \Omega(mn)$ [Indyk-Mahabadi-Vakilian'16]
- $S = \widetilde{O}(n) \Rightarrow \alpha = \Omega(n^{1/(p+1)}/p^2)$ [this work]

Final Remarks

Combinatorial optimisation: old topic
but relatively new territory for data stream algorithms

- Potential for many new research questions
- Fuller understanding of possible tradeoffs for set cover?