Optimal lower bounds for samplers, finding duplicates, and universal relation

Jelani Nelson

Harvard

March 21, 2017

joint work with Jakub Pachocki (OpenAI) and Zhengyu Wang (Harvard)

Turnstile streaming

- vector $z \in \mathbb{R}^n$ starts off as 0, updates "$z_i \leftarrow z_i + \Delta$", $\Delta \in \mathbb{R}$

Turnstile streaming

- vector $z \in \mathbb{R}^n$ starts off as 0, updates "$z_i \leftarrow z_i + \Delta$", $\Delta \in \mathbb{R}$
- data structure supporting various types of queries to z

Turnstile streaming

- vector $z \in \mathbb{R}^n$ starts off as 0, updates "$z_i \leftarrow z_i + \Delta$", $\Delta \in \mathbb{R}$
- data structure supporting various types of queries to z
- **Assumptions and examples:**
  - **Insertion-only:** $\Delta = 1$ always
    e.g. n is size of lexicon. Google search for word i causes update to i, so z_i is frequency of word i. Might want to find frequent query words ("heavy hitters").
  - **Strict turnstile:** Δ positive or negative, but $\forall i \; z_i \geq 0$ always
    e.g. graph on N vertices, $n = \binom{N}{2}$. Edge insertion of e causes $z_e \leftarrow z_e + 1$, and deletion has $\Delta = -1$. Never delete edges that don't already exist (no negative edge multiplicities).
  - **(General) turnstile:** No additional assumptions
    same as insertion-only example, but searches yesterday have $\Delta = -1$ and today have $\Delta = 1$. z_i is then change in frequency, now want to find words with large changes.

Sampling in streams

- Early work on **reservoir sampling**: sample k items from insertion-only stream using $O(k \log n)$ bits of memory.

  attributed in [Knuth'81] to Alan G. Waterman

Sampling in streams

- Early work on **reservoir sampling**: sample k items from insertion-only stream using $O(k \log n)$ bits of memory.

  attributed in [Knuth'81] to Alan G. Waterman
- solves ℓ_1-sampling in insertion-only:

  for $k = 1$, $\mathbb{P}(i \text{ is the sampled item}) = \frac{|z_i|}{\|z\|_1}$

Sampling in streams

- Early work on **reservoir sampling**: sample k items from insertion-only stream using $O(k \log n)$ bits of memory.

  attributed in [Knuth'81] to Alan G. Waterman
- solves ℓ_1-sampling in insertion-only:

  for $k = 1$, $\mathbb{P}(i \text{ is the sampled item}) = \frac{|z_i|}{\|z\|_1}$
- What about (strict) turnstile? Other sampling distributions?

Sampling in turnstile streams

ℓ_p-sampling

- $p_i = \frac{|z|_i^p}{\|z\|_p^p}$
- [Coppersmith, Kumar '04] asked whether ℓ_2 sampling is possible in small space (would lead to nearly space-optimal algorithms for ℓ_p-norm estimation for $p > 2$).
- First small-space solution in [Monemizadeh, Woodruff '10].

ℓ_0-sampling

- $p_i = \begin{cases} \frac{1}{\|z\|_0}, & z_i \neq 0 \\ 0, & \text{otherwise} \end{cases}$
- Originally asked about in [Cormode, Muthu, Rozenbaum '05] and [Frahling, Indyk, Sohler '05].
- Shown to be a useful primitive for turnstile graph streaming in [Ahn, Guha, McGregor '10].

Sampling in turnstile streams

ℓ_p-sampling $(0 < p < 2)$

*all space measured in bits

- [Monemizadeh, Woodruff '10]: in $poly(\varepsilon^{-1} \log n)$ space, whp sample has distribution within $1 \pm \varepsilon$ of $p_i = \frac{|z_i|^p}{\|z\|_p^p}$
- [Andoni, Krauthgamer, Indyk '11]: constant failure probability, $O(\varepsilon^{-p} \log^3 n)$ space for $1 \leq p \leq 2$

Sampling in turnstile streams

ℓ_p-sampling ($0 < p < 2$)

*all space measured in bits

- [Monemizadeh, Woodruff '10]: in $poly(\varepsilon^{-1} \log n)$ space, whp sample has distribution within $1 \pm \varepsilon$ of $p_i = \frac{|z_i|^p}{\|z\|_p^p}$
- [Andoni, Krauthgamer, Indyk '11]: constant failure probability, $O(\varepsilon^{-p} \log^3 n)$ space for $1 \leq p \leq 2$
- State-of-the-art. [Jowhari, Sağlam, Tardos '11]: $O(\varepsilon^{-\max\{1,p\}} \log(1/\delta) \log^2 n)$ space for $p \neq 1$. $O(\varepsilon^{-1} \log(1/\varepsilon) \log(1/\delta) \log^2 n)$ for $p = 1$.

Sampling in turnstile streams

ℓ_p-sampling $(0 < p < 2)$

*all space measured in bits

- [Monemizadeh, Woodruff '10]: in $poly(\varepsilon^{-1} \log n)$ space, whp sample has distribution within $1 \pm \varepsilon$ of $p_i = \frac{|z_i|^p}{\|z\|_p^p}$
- [Andoni, Krauthgamer, Indyk '11]: constant failure probability, $O(\varepsilon^{-p} \log^3 n)$ space for $1 \le p \le 2$
- State-of-the-art. [Jowhari, Sağlam, Tardos '11]:
  $O(\varepsilon^{-\max\{1,p\}} \log(1/\delta) \log^2 n)$ space for $p \neq 1$.
  $O(\varepsilon^{-1} \log(1/\varepsilon) \log(1/\delta) \log^2 n)$ for $p = 1$.

  for constant ε, space is $O(\log(1/\delta) \log^2 n)$.

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

- [Frahling, Indyk, Sohler'05]: $O(\log^3 n)$ space, whp success

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

- [Frahling, Indyk, Sohler'05]: $O(\log^3 n)$ space, whp success
- State-of-the-art. [Jowhari, Sağlam, Tardos '11]: $O(\log(1/\delta)\log^2 n)$ space (w.p. δ can output anything, and w.p. $1-\delta$ outputs uniformly random element from support(z))

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

- [Frahling, Indyk, Sohler'05]: $O(\log^3 n)$ space, whp success
- State-of-the-art. [Jowhari, Sağlam, Tardos '11]: $O(\log(1/\delta)\log^2 n)$ space (w.p. δ can output anything, and w.p. $1-\delta$ outputs uniformly random element from support(z))
- In fact, [JST11] spits out $\min\{\|z\|_0, \Theta(\log(1/\delta))\}$ uniform random elements from support, without replacement
- motivates studying ℓ_0-sampling$_k$ (have to output $\min\{k, \|z\|_0\}$ samples from support, w/o replacement)
- [JST11] achieves space $O(t\log^2 n)$ for ℓ_0-sampling$_k$ for $t = \max\{k, \log(1/\delta)\}$.

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

Since [Ahn, Guha, McGregor '12a], used as a subroutine in seemingly every known turnstile algorithm for dynamic graphs.

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

Since [Ahn, Guha, McGregor '12a], used as a subroutine in seemingly every known turnstile algorithm for dynamic graphs.

- connectivity [Ahn, Guha, McGregor '12a]
- *k*-connectivity [Ahn, Guha, McGregor '12a]
- bipartiteness [Ahn, Guha, McGregor '12a]
- minimum spanning tree [Ahn, Guha, McGregor '12a]
- subgraph counting [Ahn, Guha, McGregor '12b]
- minimum cut [Ahn, Guha, McGregor '12b]
- cut-sparsifiers [Ahn, Guha, McGregor '12b]
- spanners [Ahn, Guha, McGregor '12b]
- spectral sparsifiers [Ahn, Guha, McGregor '13]
- maximal matching [Chitnis et al. '15]
- maximum matching [Ahn, Guha, McGregor '1a], [Bury, Schwiegelshohn '15], [Konrad '15], [Assadi et al. '16], [Chitnis et al. '16], [Assadi et al. '17]

- vertex cover [Chitnis et al. '15], [Chitnis et al. '16]
- hitting set [Chitnis et al. '16]
- *b*-matching [Chitnis et al. '16]
- disjoint paths [Chitnis et al. '16]
- *k*-colorable subgraph and several other maximum subgraph problems [Chitnis et al. '16]
- densest subgraph [Bhattacharya et al. '15], [McGregor et al. '15], [Esfandiari et al. '16]
- vertex and hyperedge connectivity [Guha, McGregor, Tench '15]
- graph degeneracy [Farach-Colton, Tsai '16]

Sampling in turnstile streams

ℓ_0-sampling

*all space measured in bits

Since [Ahn, Guha, McGregor '12a], used as a subroutine in seemingly every known turnstile algorithm for dynamic graphs.

- connectivity [Ahn, Guha, McGregor '12a]
- k-connectivity [Ahn, Guha, McGregor '12a]
- bipartiteness [Ahn, Guha, McGregor '12a]
- minimum spanning tree [Ahn, Guha, McGregor '12a]
- subgraph counting [Ahn, Guha, McGregor '12b]
- minimum cut [Ahn, Guha, McGregor '12b]
- cut-sparsifiers [Ahn, Guha, McGregor '12b]
- spanners [Ahn, Guha, McGregor '12b]
- spectral sparsifiers [Ahn, Guha, McGregor '13]
- maximal matching [Chitnis et al. '15]
- maximum matching [Ahn, Guha, McGregor '1a], [Bury, Schwiegelshohn '15], [Konrad '15], [Assadi et al. '16], [Chitnis et al. '16], [Assadi et al. '17]

- vertex cover [Chitnis et al. '15], [Chitnis et al. '16]
- hitting set [Chitnis et al. '16]
- b-matching [Chitnis et al. '16]
- disjoint paths [Chitnis et al. '16]
- k-colorable subgraph and several other maximum subgraph problems [Chitnis et al. '16]
- densest subgraph [Bhattacharya et al. '15], [McGregor et al. '15], [Esfandiari et al. '16]
- vertex and hyperedge connectivity [Guha, McGregor, Tench '15]
- graph degeneracy [Farach-Colton, Tsai '16]

Many algs don't need ℓ_0-sample, but rather just *any* $i \in supp(z)$

One other problem: finding duplicates

Given a stream of $n+1$ integers from $[n]$, pigeonhole says there must be at least one duplicate. Find it!

One other problem: finding duplicates

Given a stream of $n+1$ integers from $[n]$, pigeonhole says there must be at least one duplicate. Find it!

- [Gopalan, Radhakrishnan '09]: $O(\log^3 n)$ space for constant δ (reduction to ℓ_1-sampling)

One other problem: finding duplicates

Given a stream of $n+1$ integers from $[n]$, pigeonhole says there must be at least one duplicate. Find it!

- [Gopalan, Radhakrishnan '09]: $O(\log^3 n)$ space for constant δ (reduction to ℓ_1-sampling)
- State-of-the-art. [Jowhari, Sağlam, Tardos '11]: $O(\log(1/\delta)\log^2 n)$ space for failure prob. δ.

Our main contribution

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of *support*(z) in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of *support*(z) in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.
- Finding *any* k elements from support in strict turnstile requires $\Omega(\min\{n, t\log^2(n/t)\})$ space for $t = \max\{k, \log(1/\delta)\}$.

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of $\mathit{support}(z)$ in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.
- Finding *any* k elements from support in strict turnstile requires $\Omega(\min\{n, t\log^2(n/t)\})$ space for $t = \max\{k, \log(1/\delta)\}$.
- Implies tight lower bounds for ℓ_p-sampling for any $0 \leq p < 2$ and also ℓ_0-sampling$_k$ for $t < n^{.99}$.

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of $\mathit{support}(z)$ in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.
- Finding *any* k elements from support in strict turnstile requires $\Omega(\min\{n, t\log^2(n/t)\})$ space for $t = \max\{k, \log(1/\delta)\}$.
- Implies tight lower bounds for ℓ_p-sampling for any $0 \le p < 2$ and also ℓ_0-sampling$_k$ for $t < n^{.99}$.
- Also show lower bound (tight for $\delta > 2^{-n^{.99}}$) of $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space for finding duplicates.

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of $\mathit{support}(z)$ in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.
- Finding *any* k elements from support in strict turnstile requires $\Omega(\min\{n, t\log^2(n/t)\})$ space for $t = \max\{k, \log(1/\delta)\}$.
- Implies tight lower bounds for ℓ_p-sampling for any $0 \le p < 2$ and also ℓ_0-sampling$_k$ for $t < n^{.99}$.
- Also show lower bound (tight for $\delta > 2^{-n^{.99}}$) of $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space for finding duplicates.
- Lower bounds from **UR** (**universal relation**), as in [JST11]

heart of our new tight result: new tight lower bound for **UR**

Our contribution [Nelson, Pachocki, Wang '17]

- Finding *any* element of $\mathit{support}(z)$ in strict turnstile streams requires $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space.
- Finding *any* k elements from support in strict turnstile requires $\Omega(\min\{n, t\log^2(n/t)\})$ space for $t = \max\{k, \log(1/\delta)\}$.
- Implies tight lower bounds for ℓ_p-sampling for any $0 \le p < 2$ and also ℓ_0-sampling$_k$ for $t < n^{.99}$.
- Also show lower bound (tight for $\delta > 2^{-n^{.99}}$) of $\Omega(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$ space for finding duplicates.
- Lower bounds from **UR** (**universal relation**), as in [JST11]

  heart of our new tight result: new tight lower bound for **UR**
- **Theorem:** $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}) = \Theta(\min\{n, \log(1/\delta)\log^2 \frac{n}{\log(1/\delta)}\})$

Universal relation

- Arose out of work of [Karchmer, Wigderson '88] on depth lower bounds for circuits
  - $f : \{0,1\}^n \to \{0,1\}$
  - Alice receives $x \in f^{-1}(0)$, Bob receives $y \in f^{-1}(1)$ (so $x \neq y$)
  - must find $i \in [n]$ such that $x_i \neq y_i$

Universal relation

- Arose out of work of [Karchmer, Wigderson '88] on depth lower bounds for circuits
  - $f : \{0,1\}^n \to \{0,1\}$
  - Alice receives $x \in f^{-1}(0)$, Bob receives $y \in f^{-1}(1)$ (so $x \neq y$)
  - must find $i \in [n]$ such that $x_i \neq y_i$
  - **Thm [KW88]**: $D(f) = C(f)$
    (Depth equals Deterministic Communication Complexity)

Universal relation

- Arose out of work of [Karchmer, Wigderson '88] on depth lower bounds for circuits
  - $f : \{0,1\}^n \to \{0,1\}$
  - Alice receives $x \in f^{-1}(0)$, Bob receives $y \in f^{-1}(1)$ (so $x \neq y$)
  - must find $i \in [n]$ such that $x_i \neq y_i$
  - **Thm [KW88]**: $D(f) = C(f)$
    (Depth equals Deterministic Communication Complexity)
  - Used in [KW88] to obtain tight $\Omega(\log^2 n)$ depth lower bound for monotone circuits computing s-t connectivity

Universal relation

- Arose out of work of [Karchmer, Wigderson '88] on depth lower bounds for circuits
  - $f : \{0,1\}^n \to \{0,1\}$
  - Alice receives $x \in f^{-1}(0)$, Bob receives $y \in f^{-1}(1)$ (so $x \neq y$)
  - must find $i \in [n]$ such that $x_i \neq y_i$
  - **Thm [KW88]**: $D(f) = C(f)$
    (Depth equals Deterministic Communication Complexity)
  - Used in [KW88] to obtain tight $\Omega(\log^2 n)$ depth lower bound for monotone circuits computing s-t connectivity
- Later, [Karchmer, Raz, Wigderson'91] outlined strategy to separate $\mathbf{NC}^1$ from $\mathbf{P}$ (and even from $\mathbf{NC}^2$): show a form of direct sum theorem for "k-fold composition" of functions ("KRW conjecture"), then apply k-fold composition to a "hard" function on $\log n$ variables with $k = \log n / \log\log n$.

Universal relation

- Arose out of work of [Karchmer, Wigderson '88] on depth lower bounds for circuits
  - $f : \{0,1\}^n \to \{0,1\}$
  - Alice receives $x \in f^{-1}(0)$, Bob receives $y \in f^{-1}(1)$ (so $x \neq y$)
  - must find $i \in [n]$ such that $x_i \neq y_i$
  - **Thm [KW88]**: $D(f) = C(f)$
    (Depth equals Deterministic Communication Complexity)
  - Used in [KW88] to obtain tight $\Omega(\log^2 n)$ depth lower bound for monotone circuits computing s-t connectivity
- Later, [Karchmer, Raz, Wigderson'91] outlined strategy to separate $\mathbf{NC}^1$ from $\mathbf{P}$ (and even from $\mathbf{NC}^2$): show a form of direct sum theorem for "k-fold composition" of functions ("KRW conjecture"), then apply k-fold composition to a "hard" function on $\log n$ variables with $k = \log n / \log \log n$.
- **Warmup [KRW91]:** prove that direct sum theorem holds for k-fold composition of **UR** relation. (was later resolved positively in [Edmonds, Impagliazzo, Rudich, Sgall '91])

Universal relation

- **UR**: forget about the function f, just promised that $x \neq y$
- Alice, Bob get $x, y \in \{0, 1\}^n$ (resp.) **with promise** $x \neq y$
- must find $i \in [n]$ such that $x_i \neq y_i$

Universal relation

- **UR**: forget about the function f, just promised that $x \neq y$
- Alice, Bob get $x, y \in \{0,1\}^n$ (resp.) **with promise** $x \neq y$
- must find $i \in [n]$ such that $x_i \neq y_i$
- Deterministic comm. complexity of **UR** *very* well understood (upper and lower bounds off by an additive 3 bits!), even in bounded number of rounds [Tardos, Zwick '97]

Universal relation

- **UR**: forget about the function f, just promised that $x \neq y$
- Alice, Bob get $x, y \in \{0,1\}^n$ (resp.) **with promise $x \neq y$**
- must find $i \in [n]$ such that $x_i \neq y_i$
- Deterministic comm. complexity of **UR** *very* well understood (upper and lower bounds off by an additive 3 bits!), even in bounded number of rounds [Tardos, Zwick '97]
- Here we focus on one-way communication complexity in the public coin model, $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR})$:
  - Alice sends a single message to Bob
  - Bob, based on that message, must output $i \in [n]$ s.t. $\mathbb{P}(x_i \neq y_i) \geq 1 - \delta$

Universal relation

- **UR**: forget about the function f, just promised that $x \neq y$
- Alice, Bob get $x, y \in \{0,1\}^n$ (resp.) **with promise** $x \neq y$
- must find $i \in [n]$ such that $x_i \neq y_i$
- Deterministic comm. complexity of **UR** *very* well understood (upper and lower bounds off by an additive 3 bits!), even in bounded number of rounds [Tardos, Zwick '97]
- Here we focus on one-way communication complexity in the public coin model, $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR})$:
  - Alice sends a single message to Bob
  - Bob, based on that message, must output $i \in [n]$ s.t. $\mathbb{P}(x_i \neq y_i) \geq 1 - \delta$
- will also look at some variants / promise versions:
  - $\mathbf{UR}_k$: Bob must output $\min\{k, \|x - y\|_0\}$ differing indices
  - $\mathbf{UR}^{\subset}$: Alice is promised $supp(y) \subsetneq supp(x)$
  - $\mathbf{UR}^{+}$: Bob knows $|supp(x)|$ (not super important ...)

Universal relation

Thm [NPW'17]: For any δ bounded away from 1 and any $k \in [n]$, $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}_k) = \Theta(\min\{n, t\log^2(n/t)\})$ for $t = \max\{k, \log(1/\delta)\}$.

Universal relation

Thm [NPW'17]: For any δ bounded away from 1 and any $k \in [n]$, $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}_k) = \Theta(\min\{n, t\log^2(n/t)\})$ for $t = \max\{k, \log(1/\delta)\}$.

*In fact, lower bound even holds for the special case $\mathbf{UR}_k^{\subset,+}$

Universal relation

Thm [NPW'17]: For any δ bounded away from 1 and any $k \in [n]$, $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}_k) = \Theta(\min\{n, t\log^2(n/t)\})$ for $t = \max\{k, \log(1/\delta)\}$.

*In fact, lower bound even holds for the special case $\mathbf{UR}_k^{\subset,+}$

Upper bound is a slight improvement of [JST11], which showed $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}_k) = O(\min\{n, t\log^2 n\})$.

Relevance to streaming lower bounds

[JST11] reduced **UR** to finding duplicates and (general turnstile) ℓ_p-sampling, then showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}) = \Omega(\log^2 n)$.

Relevance to streaming lower bounds

[JST11] reduced **UR** to finding duplicates and (general turnstile) ℓ_p-sampling, then showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}) = \Omega(\log^2 n)$.

In fact [JST11] even showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}^{\subset}) = \Omega(\log^2 n)$ (via reduction from Augmented-Indexing [Miltersen et al. '98], [Ergün, Jowhari, Sağlam '10], [Jayram, Woodruff '11]).

Relevance to streaming lower bounds

[JST11] reduced **UR** to finding duplicates and (general turnstile) ℓ_p-sampling, then showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}) = \Omega(\log^2 n)$.

In fact [JST11] even showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}^{\subset}) = \Omega(\log^2 n)$ (via reduction from Augmented-Indexing [Miltersen et al. '98], [Ergün, Jowhari, Sağlam '10], [Jayram, Woodruff '11]).

- This observation makes reductions simpler and more powerful (hardness for even strict turnstile, and finding any element in the support instead of ℓ_p-sampling).

Relevance to streaming lower bounds

[JST11] reduced **UR** to finding duplicates and (general turnstile) ℓ_p-sampling, then showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}) = \Omega(\log^2 n)$.

In fact [JST11] even showed $\mathbf{R}_\delta^{pub,\rightarrow}(\mathbf{UR}^{\subset}) = \Omega(\log^2 n)$ (via reduction from Augmented-Indexing [Miltersen et al. '98], [Ergün, Jowhari, Sağlam '10], [Jayram, Woodruff '11]).

- This observation makes reductions simpler and more powerful (hardness for even strict turnstile, and finding any element in the support instead of ℓ_p-sampling).
- It seems [JST11] not realize that they proved this (or at least, they did not realize that having proved this makes reductions a tad simpler!).

Reductions from $\mathbf{UR}^{\subset}$

Claim: Space complexity of finding an element in $supp(z)$ in strict turnstile with failure probability δ is at least $\mathbf{R}_{\delta}^{\rightarrow, pub}(\mathbf{UR}^{\subset})$.

Reductions from $\mathbf{UR}^{\subset}$

Claim: Space complexity of finding an element in $supp(z)$ in strict turnstile with failure probability δ is at least $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset})$.
Proof: Reduction from $\mathbf{UR}^{\subset}$. Suppose $\mathcal{A}$ is algorithm for streaming problem. Alice updates $z_i \leftarrow z_i + 1$ for all $i \in supp(x)$ then sends memory contents of $\mathcal{A}$ to Bob as message. Bob continues running $\mathcal{A}$ and does $z_i \leftarrow z_i - 1$ for all $i \in supp(y)$. Then Bob outputs $\mathcal{A}.\mathtt{query}()$.

Reductions from $\mathbf{UR}^{\subset}$

Claim: Space complexity of finding duplicate in stream of length $n+1$ with failure probability δ is at least $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.

Reductions from $\mathbf{UR}^{\subset}$

Claim: Space complexity of finding duplicate in stream of length $n+1$ with failure probability δ is at least $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.
Proof: Reduction from $\mathbf{UR}^{\subset,+}$. Suppose $\mathcal{A}$ is algorithm for finding duplicate. Alice puts i in stream for each $i \in supp(x)$ then sends memory contents of $\mathcal{A}$ to Bob as message. Bob continues running $\mathcal{A}$ by appending to stream $n+1-|supp(x)|$ indices $i \in [n] \setminus supp(y)$. Then Bob outputs $\mathcal{A}.\mathtt{query}()$.

The Main Event

Proof of our new lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Lower bound plan

- **Idea:** if $\mathcal{P}$ is efficient 1-way protocol for $\mathbf{UR}^{\subset,+}$, use it to design efficient Las Vegas encoding for $\binom{[n]}{m}$ for particular m (encoding length is random variable; decoder always succeeds)

Lower bound plan

- **Idea:** if $\mathcal{P}$ is efficient 1-way protocol for $\mathbf{UR}^{\subset,+}$, use it to design efficient Las Vegas encoding for $\binom{[n]}{m}$ for particular m (encoding length is random variable; decoder always succeeds)
- any such encoding scheme needs $\geq \lg\binom{n}{m} = \Omega(m \log(n/m))$ bits in expectation $\Longrightarrow$ lower bound for $\mathcal{P}$

Lower bound plan

- **Idea:** if $\mathcal{P}$ is efficient 1-way protocol for $\mathbf{UR}^{\subset,+}$, use it to design efficient Las Vegas encoding for $\binom{[n]}{m}$ for particular m (encoding length is random variable; decoder always succeeds)
- any such encoding scheme needs $\geq \lg\binom{n}{m} = \Omega(m \log(n/m))$ bits in expectation $\Longrightarrow$ lower bound for $\mathcal{P}$
- **Notation:**
  - E: encoder
  - D: decoder
  - Alice: 1^{st} player in supposed efficient protocol $\mathcal{P}$ for $\mathbf{UR}^{\subset,+}$
  - Bob: 2^{nd} player in supposed efficient protocol $\mathcal{P}$ for $\mathbf{UR}^{\subset,+}$
  - S: subset of $[n]$, $|S| = m$, to be encoded
  - $\mathbf{1}_S \in \{0,1\}^n$ is indicator vector of S

Lower bound plan

- **Idea:** if $\mathcal{P}$ is efficient 1-way protocol for $\mathbf{UR}^{\subset,+}$, use it to design efficient Las Vegas encoding for $\binom{[n]}{m}$ for particular m (encoding length is random variable; decoder always succeeds)
- any such encoding scheme needs $\geq \lg\binom{n}{m} = \Omega(m\log(n/m))$ bits in expectation $\implies$ lower bound for $\mathcal{P}$
- **Notation:**
  - E: encoder
  - D: decoder
  - Alice: 1^{st} player in supposed efficient protocol $\mathcal{P}$ for $\mathbf{UR}^{\subset,+}$
  - Bob: 2^{nd} player in supposed efficient protocol $\mathcal{P}$ for $\mathbf{UR}^{\subset,+}$
  - S: subset of $[n]$, $|S| = m$, to be encoded
  - $\mathbf{1}_S \in \{0,1\}^n$ is indicator vector of S
  - The $+$ in $\mathbf{UR}^{\subset,+}$ will mean E/D both know m (not a big deal: otherwise E could write m down)

Simple lower bound

$E(S)$ is Alice's message $M \in \{0,1\}^s$ to Bob on input $x = \mathbf{1}_S$.

Simple lower bound

$E(S)$ is Alice's message $M \in \{0,1\}^s$ to Bob on input $x = \mathbf{1}_S$.

procedure $D(M)$
    $T \leftarrow \emptyset$
    for $r = 1, \ldots, m$ **do**
        Let i be Bob's output upon receiving message M from Alice when Bob's input is $\mathbf{1}_T$
        $T \leftarrow T \cup \{i\}$
    end for
    return T
end procedure

Simple lower bound

$E(S)$ is Alice's message $M \in \{0,1\}^s$ to Bob on input $x = \mathbf{1}_S$.

---

procedure $D(M)$
  $T \leftarrow \emptyset$
  for $r = 1, \ldots, m$ **do**
    Let i be Bob's output upon receiving message M from Alice when Bob's input is $\mathbf{1}_T$
    $T \leftarrow T \cup \{i\}$
  end for
  return T
end procedure

---

*This is, hopefully, a Monte Carlo encoding/decoding scheme

Want $\mathbb{P}(T = S)$ to be large (at least 1/2, say)

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\implies$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\implies$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$
  $\implies$ can set $m = n/2$ and get $s = \Omega(n)$ for $\delta < 1/n$

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\implies$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$
  $\implies$ can set $m = n/2$ and get $s = \Omega(n)$ for $\delta < 1/n$
- **Problem:** There's an $O(\log^3 n)$ *upper bound* for $\delta = \frac{1}{poly(n)}$
  (Alice sends memory of ℓ_0-sampler sketch to Bob, run on $\mathbf{1}_S$)

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\implies$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$
  $\implies$ can set $m = n/2$ and get $s = \Omega(n)$ for $\delta < 1/n$
- **Problem:** There's an $O(\log^3 n)$ *upper bound* for $\delta = \frac{1}{poly(n)}$ (Alice sends memory of ℓ_0-sampler sketch to Bob, run on $\mathbf{1}_S$)
- Problem is even worse: $E(S)$ could have first applied error-correcting code to $\mathbf{1}_S$ to obtain $S' \in [\Theta(n)]$, then Bob could recover S with good probability even for δ a constant! But for constant δ, there's $O(\log^2 n)$ upper bound for $\mathbf{UR}^{\subset,+}$.

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\implies$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$
  $\implies$ can set $m = n/2$ and get $s = \Omega(n)$ for $\delta < 1/n$
- **Problem:** There's an $O(\log^3 n)$ *upper bound* for $\delta = \frac{1}{poly(n)}$
  (Alice sends memory of ℓ_0-sampler sketch to Bob, run on $\mathbf{1}_S$)
- Problem is even worse: $E(S)$ could have first applied error-correcting code to $\mathbf{1}_S$ to obtain $S' \in [\Theta(n)]$, then Bob could recover S with good probability even for δ a constant! But for constant δ, there's $O(\log^2 n)$ upper bound for $\mathbf{UR}^{\subset,+}$.
- **What went wrong here?**

(Wrong) Analysis: take 1

- Original failure probability of $\mathcal{P}$ is δ
  $\Longrightarrow$ failure probability of decoder is $\delta m < 1/2$ for $\delta < \frac{1}{2m}$
  $\Longrightarrow$ can set $m = n/2$ and get $s = \Omega(n)$ for $\delta < 1/n$
- **Problem:** There's an $O(\log^3 n)$ *upper bound* for $\delta = \frac{1}{poly(n)}$ (Alice sends memory of ℓ_0-sampler sketch to Bob, run on $\mathbf{1}_S$)
- Problem is even worse: $E(S)$ could have first applied error-correcting code to $\mathbf{1}_S$ to obtain $S' \in [\Theta(n)]$, then Bob could recover S with good probability even for δ a constant! But for constant δ, there's $O(\log^2 n)$ upper bound for $\mathbf{UR}^{\subset,+}$.
- **What went wrong here?**
- **Adaptivity!!!**
- Correctness of $\mathcal{P}$ says $\forall x, y,\ \mathbb{P}(\mathcal{P}$ succeeds on $x, y) \geq 1 - \delta$. **Bob not allowed to choose** y **based on** $\mathcal{P}$**'s random coins**.

Correct Analysis

- Fix S and define event $\mathcal{E}_T$: $\mathcal{P}$ succeeds when $x = \mathbf{1}_S$, $y = \mathbf{1}_T$.

Correct Analysis

- Fix S and define event $\mathcal{E}_T$: $\mathcal{P}$ succeeds when $x = \mathbf{1}_S$, $y = \mathbf{1}_T$.
- If $\bigcap_{T \subsetneq S} \mathcal{E}_T$ occurs, then decoder succeeds.

Correct Analysis

- Fix S and define event $\mathcal{E}_T$: $\mathcal{P}$ succeeds when $x = \mathbf{1}_S$, $y = \mathbf{1}_T$.
- If $\bigcap_{T \subsetneq S} \mathcal{E}_T$ occurs, then decoder succeeds.
- $$\mathbb{P}(\neg(\bigcap_{T \subsetneq S} \mathcal{E}_T)) = \mathbb{P}(\bigcup_{T \subsetneq S} \overline{\mathcal{E}_T}) < \delta 2^m.$$

  set $m = \lfloor \lg(1/\delta) \rfloor - 1$, so decoder succeeds w.p. $> 1/2$

Correct Analysis

- Fix S and define event $\mathcal{E}_T$: $\mathcal{P}$ succeeds when $x = \mathbf{1}_S$, $y = \mathbf{1}_T$.
- If $\bigcap_{T \subsetneq S} \mathcal{E}_T$ occurs, then decoder succeeds.
- $$\mathbb{P}(\neg(\bigcap_{T \subsetneq S} \mathcal{E}_T)) = \mathbb{P}(\bigcup_{T \subsetneq S} \overline{\mathcal{E}_T}) < \delta 2^m.$$

  set $m = \lfloor \lg(1/\delta) \rfloor - 1$, so decoder succeeds w.p. $> 1/2$
- $\implies s = |M| = \Omega(m \log(n/m)) = \Omega(\log(1/\delta) \log \frac{n}{\log(1/\delta)})$

Correct Analysis

- Fix S and define event $\mathcal{E}_T$: $\mathcal{P}$ succeeds when $x = \mathbf{1}_S$, $y = \mathbf{1}_T$.
- If $\bigcap_{T \subsetneq S} \mathcal{E}_T$ occurs, then decoder succeeds.
- $$\mathbb{P}(\neg(\bigcap_{T \subsetneq S} \mathcal{E}_T)) = \mathbb{P}(\bigcup_{T \subsetneq S} \overline{\mathcal{E}_T}) < \delta 2^m.$$

  set $m = \lfloor \lg(1/\delta) \rfloor - 1$, so decoder succeeds w.p. $> 1/2$
- $\implies s = |M| = \Omega(m \log(n/m)) = \Omega(\log(1/\delta) \log \frac{n}{\log(1/\delta)})$
- to get optimal lower bound, need another $\log \frac{n}{\log(1/\delta)}$ factor

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Moral of our work: it's ok to make adaptive queries to mechanism that are not independent of the randomness of the mechanism, **if the amount of dependence can be controlled**

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Moral of our work: it's ok to make adaptive queries to mechanism that are not independent of the randomness of the mechanism, **if the amount of dependence can be controlled**

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y) = 1) \leq \delta$ where $0 < \delta < 1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y) = 1) \leq \frac{I(X;Y) + 1}{\log \frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Optimal lower bound for $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Moral of our work: it's ok to make adaptive queries to mechanism that are not independent of the randomness of the mechanism, **if the amount of dependence can be controlled**

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y)=1) \le \delta$ where $0<\delta<1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log\frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Interpretation: Fix input x to Alice. X is internal randomness of $\mathcal{P}$, and $f(x,y)$ is 1 iff $\mathcal{P}$ is incorrect when Bob has input y.

Adaptivity lemma

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y)=1) \le \delta$ where $0<\delta<1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log\frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Adaptivity lemma

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y) = 1) \le \delta$ where $0 < \delta < 1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y) = 1) \le \frac{I(X;Y) + 1}{\log \frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Is the above lemma tight?

Yes. $x, y \in [n]$, X is uniform. $f(x,y) = 1$ iff $x = y$. $\delta = \frac{1}{n}$.

Adaptivity lemma

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y)=1) \le \delta$ where $0<\delta<1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log\frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Is the above lemma tight?

Yes. $x, y \in [n]$, X is uniform. $f(x,y)=1$ iff $x=y$. $\delta = \frac{1}{n}$.

- consider this Y: equals X w.p. $\frac{t}{\log n}$, and otherwise is uniform

Adaptivity lemma

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y)=1) \le \delta$ where $0<\delta<1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log\frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Is the above lemma tight?

Yes. $x,y\in[n]$, X is uniform. $f(x,y)=1$ iff $x=y$. $\delta=\frac{1}{n}$.

- consider this Y: equals X w.p. $\frac{t}{\log n}$, and otherwise is uniform
- $I(X;Y)=t$

Adaptivity lemma

Lemma [NPW'17]: Consider $f\colon \{0,1\}^b \times \{0,1\}^q \to \{0,1\}$ and $X \in \{0,1\}^b$ uniformly random. If $\forall y \in \{0,1\}^q,\ \mathbb{P}(f(X,y)=1) \le \delta$ where $0 < \delta < 1$, then for any random variable Y supported on $\{0,1\}^q$,

$$\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log \frac{1}{\delta}},$$

where $I(X;Y)$ is the mutual information between X and Y.

Is the above lemma tight?

Yes. $x, y \in [n]$, X is uniform. $f(x,y)=1$ iff $x=y$. $\delta = \frac{1}{n}$.

- consider this Y: equals X w.p. $\frac{t}{\log n}$, and otherwise is uniform
- $I(X;Y) = t$
- $\mathbb{P}(f(X,Y)=1) = \frac{t}{\log n}\cdot 1 + (1-\frac{t}{\log n})\cdot \frac{1}{n} \approx \frac{t}{\log n}$

Rest of talk

1. Proving the lemma (short).
2. Using the lemma to lower bound $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.

Rest of talk

1. Proving the lemma (short).
2. Using the lemma to lower bound $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.

Proof of lemma

Lemma: $\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log \frac{1}{\delta}}$

Proof of lemma

Lemma: $\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log\frac{1}{\delta}}$

- Equivalent to prove $I(X;Y) \ge (\mathbb{E}\, f(X,Y)) \cdot \log\frac{1}{\delta} - 1$
- $I(X;Y) = H(X) - H(X|Y) = b - H(X|Y)$.

  Want to upper bound $H(X|Y)$.

Proof of lemma

Lemma: $\mathbb{P}(f(X, Y) = 1) \leq \frac{I(X;Y)+1}{\log \frac{1}{\delta}}$

- Equivalent to prove $I(X; Y) \geq (\mathbb{E}\, f(X, Y)) \cdot \log \frac{1}{\delta} - 1$
- $I(X; Y) = H(X) - H(X|Y) = b - H(X|Y)$.

  Want to upper bound $H(X|Y)$.
- Consider communication problem: Alice gets X, Y, Bob only gets Y. Expected number of bits Alice needs to send Bob so he can recover X with probability 1 is exactly $H(X|Y)$.

Proof of lemma

Lemma: $\mathbb{P}(f(X,Y)=1) \leq \frac{I(X;Y)+1}{\log\frac{1}{\delta}}$

- Equivalent to prove $I(X;Y) \geq (\mathbb{E}\, f(X,Y)) \cdot \log\frac{1}{\delta} - 1$
- $I(X;Y) = H(X) - H(X|Y) = b - H(X|Y)$.

  Want to upper bound $H(X|Y)$.
- Consider communication problem: Alice gets X, Y, Bob only gets Y. Expected number of bits Alice needs to send Bob so he can recover X with probability 1 is exactly $H(X|Y)$.
- A cheap protocol: Alice sends $f(X,Y)$ (1 bit). If $f(X,Y)=0$, also sends all of X (b bits). Else sends index of X in $\{x : f(x,Y)=1\}$ ($\log(\delta 2^b) = b - \log\frac{1}{\delta}$ bits).

Proof of lemma

Lemma: $\mathbb{P}(f(X,Y)=1) \le \frac{I(X;Y)+1}{\log \frac{1}{\delta}}$

- Equivalent to prove $I(X;Y) \ge (\mathbb{E}\, f(X,Y)) \cdot \log \frac{1}{\delta} - 1$
- $I(X;Y) = H(X) - H(X|Y) = b - H(X|Y)$.

  Want to upper bound $H(X|Y)$.
- Consider communication problem: Alice gets X, Y, Bob only gets Y. Expected number of bits Alice needs to send Bob so he can recover X with probability 1 is exactly $H(X|Y)$.
- A cheap protocol: Alice sends $f(X,Y)$ (1 bit). If $f(X,Y)=0$, also sends all of X (b bits). Else sends index of X in $\{x : f(x,Y)=1\}$ ($\log(\delta 2^b) = b - \log \frac{1}{\delta}$ bits).

  $\implies H(X|Y) \le 1 + b - (\mathbb{E}\, f(X,Y)) \cdot \log \frac{1}{\delta}$ as desired.

Rest of talk

1. Proving the lemma (short).
2. Using the lemma to lower bound $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.

Rest of talk

1. Proving the lemma (short).
2. Using the lemma to lower bound $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$.

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S \backslash A$ explicitly written down ($\log\binom{n}{|B|}$ bits).

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S \backslash A$ explicitly written down ($\log\binom{n}{|B|}$ bits).
- A comes from $R = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ iterations in decoder. Will have $\mathcal{P}$ succeeding in $\frac{R}{2}$ iterations in expectation.

Optimal lower bound for $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log \frac{1}{\delta} \log \frac{n}{\log \frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S \backslash A$ explicitly written down ($\log \binom{n}{|B|}$ bits).
- A comes from $R = \Theta(\log \frac{1}{\delta} \log \frac{n}{\log \frac{1}{\delta}})$ iterations in decoder. Will have $\mathcal{P}$ succeeding in $\frac{R}{2}$ iterations in expectation.
- In light of Lemma, D will pretend to be Bob in each of the R iterations such that for all $j \in [R]$, y_j in iteration j has mutual information $\leq \frac{1}{2} \log \frac{1}{\delta} - 1$ with the randomness used by $\mathcal{P}$.

Optimal lower bound for $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S \backslash A$ explicitly written down ($\log\binom{n}{|B|}$ bits).
- A comes from $R = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ iterations in decoder. Will have $\mathcal{P}$ succeeding in $\frac{R}{2}$ iterations in expectation.
- In light of Lemma, D will pretend to be Bob in each of the R iterations such that for all $j \in [R]$, y_j in iteration j has mutual information $\leq \frac{1}{2}\log\frac{1}{\delta} - 1$ with the randomness used by $\mathcal{P}$.
- After iteration j, D randomly adds t_j elements of B to T to dilute info about elements of S recovered from M so far.

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S\backslash A$ explicitly written down ($\log\binom{n}{|B|}$ bits).
- A comes from $R = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ iterations in decoder. Will have $\mathcal{P}$ succeeding in $\frac{R}{2}$ iterations in expectation.
- In light of Lemma, D will pretend to be Bob in each of the R iterations such that for all $j \in [R]$, y_j in iteration j has mutual information $\leq \frac{1}{2}\log\frac{1}{\delta} - 1$ with the randomness used by $\mathcal{P}$.
- After iteration j, D randomly adds t_j elements of B to T to dilute info about elements of S recovered from M so far.
- Need t_j big enough to get enough information dilution. This forces $R = O(\log\frac{1}{\delta}\log\frac{m}{\log\frac{1}{\delta}})$.

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

- **Our approach:** Give up on D recovering all of S from M.
- D will recover subset $A \subset S$, $\mathbb{E}\,|A| = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ from M. $E(S)$ then is the concatenation of M, together with the elements $B = S \backslash A$ explicitly written down ($\log\binom{n}{|B|}$ bits).
- A comes from $R = \Theta(\log\frac{1}{\delta}\log\frac{n}{\log\frac{1}{\delta}})$ iterations in decoder. Will have $\mathcal{P}$ succeeding in $\frac{R}{2}$ iterations in expectation.
- In light of Lemma, D will pretend to be Bob in each of the R iterations such that for all $j \in [R]$, y_j in iteration j has mutual information $\leq \frac{1}{2}\log\frac{1}{\delta} - 1$ with the randomness used by $\mathcal{P}$.
- After iteration j, D randomly adds t_j elements of B to T to dilute info about elements of S recovered from M so far.
- Need t_j big enough to get enough information dilution. This forces $R = O(\log\frac{1}{\delta}\log\frac{m}{\log\frac{1}{\delta}})$.
- Will get lower bound $|M| = \Omega(R\lg\frac{n}{m}) = \Omega(\lg\frac{1}{\delta}\lg\frac{m}{\lg\frac{1}{\delta}}\lg\frac{n}{m})$

  set $m = \sqrt{n\log\frac{1}{\delta}}$

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Variables shared by E and D.

$m \leftarrow \lfloor \sqrt{n \log \frac{1}{\delta}} \rfloor$
$K \leftarrow \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$
$R \leftarrow \lfloor K \log(m/4K) \rfloor$
for $r = 0, \ldots, R$ **do**
    $n_r \leftarrow \lfloor m \cdot 2^{-\frac{r}{K}} \rfloor$ $\triangleright$ $|S_r| = n_r$, and $\forall r\ n_r - n_{r+1} \geq 2$
end for
π is a random permutation on $[n]$

Optimal lower bound for $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Variables shared by E and D.

$m \leftarrow \lfloor\sqrt{n \log \frac{1}{\delta}}\rfloor$
$K \leftarrow \lfloor\frac{1}{16}\log\frac{1}{\delta}\rfloor$
$R \leftarrow \lfloor K \log(m/4K)\rfloor$
for $r = 0, \ldots, R$ **do**
    $n_r \leftarrow \lfloor m \cdot 2^{-\frac{r}{K}}\rfloor$ $\quad\triangleright\ |S_r| = n_r$, and $\forall r\ n_r - n_{r+1} \geq 2$
end for
π is a random permutation on $[n]$

---

n_j is such that after j iterations, D has already recovered $m - n_j$ elements of S (S_j, $|S_j| = n_j$, remains to be recovered)

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$

$\triangleright$ M is $\mathrm{Alice}(\mathbf{1}_S)$

$\triangleright$ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds

$\triangleright$ B contains all elements of S that D doesn't recover via M

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$
  ▷ M is $\mathrm{Alice}(\mathbf{1}_S)$
  ▷ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds
  ▷ B contains all elements of S that D doesn't recover via M
  $A \leftarrow \emptyset$ ▷ the subset of S we recover just from M
  $T_0 \leftarrow \emptyset$ ▷ subset of S we've built up so far

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$
  ▷ M is $\mathrm{Alice}(\mathbf{1}_S)$
  ▷ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds
  ▷ B contains all elements of S that D doesn't recover via M
  $A \leftarrow \emptyset$ ▷ the subset of S we recover just from M
  $T_0 \leftarrow \emptyset$ ▷ subset of S we've built up so far
  for $r = 1, \ldots, R$ **do** ▷ each iteration tries to recover 1 elt via M
    $T_r \leftarrow T_{r-1}$

Optimal lower bound for $\mathbf{R}_\delta^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$

  ▷ M is $\mathrm{Alice}(\mathbf{1}_S)$

  ▷ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds

  ▷ B contains all elements of S that D doesn't recover via M

  $A \leftarrow \emptyset$ ▷ the subset of S we recover just from M

  $T_0 \leftarrow \emptyset$ ▷ subset of S we've built up so far

  for $r = 1, \ldots, R$ **do** ▷ each iteration tries to recover 1 elt via M

    $T_r \leftarrow T_{r-1}$

    if $b_r = 1$ **then** ▷ this means Bob succeeds in round r

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

procedure $D(M, B, b)$
    ▷ M is $\mathrm{Alice}(\mathbf{1}_S)$
    ▷ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds
    ▷ B contains all elements of S that D doesn't recover via M
    $A \leftarrow \emptyset$ ▷ the subset of S we recover just from M
    $T_0 \leftarrow \emptyset$ ▷ subset of S we've built up so far
    for $r = 1, \ldots, R$ **do** ▷ each iteration tries to recover 1 elt via M
        $T_r \leftarrow T_{r-1}$
        if $b_r = 1$ **then** ▷ this means Bob succeeds in round r
            $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{T_{r-1}})$ ▷ Invariant: $T_r = S \backslash S_r$
            $A \leftarrow A \cup \{s_r\}$, $T_r \leftarrow T_r \cup \{s_r\}$
        end if

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$

 ▷ M is $\mathrm{Alice}(\mathbf{1}_S)$

 ▷ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds

 ▷ B contains all elements of S that D doesn't recover via M

 $A \leftarrow \emptyset$ ▷ the subset of S we recover just from M

 $T_0 \leftarrow \emptyset$ ▷ subset of S we've built up so far

 for $r = 1, \ldots, R$ **do** ▷ each iteration tries to recover 1 elt via M

  $T_r \leftarrow T_{r-1}$

  if $b_r = 1$ **then** ▷ this means Bob succeeds in round r

   $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{T_{r-1}})$ ▷ Invariant: $T_r = S \backslash S_r$

   $A \leftarrow A \cup \{s_r\}$, $T_r \leftarrow T_r \cup \{s_r\}$

  end if

  Insert $m - n_r - |T_r|$ items from $B \backslash T_r$ into T_r with smallest π_i

  ▷ **"Differential Privacy"** step. Still n_r elements left to recover.

 end for

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Decoding algorithm to recover $S \subset [n]$, $|S| = m$

---

procedure $D(M, B, b)$
  $\triangleright$ M is $\mathrm{Alice}(\mathbf{1}_S)$
  $\triangleright$ $b \in \{0,1\}^R$ indicates rounds in which Bob succeeds
  $\triangleright$ B contains all elements of S that D doesn't recover via M
  $A \leftarrow \emptyset$ $\triangleright$ the subset of S we recover just from M
  $T_0 \leftarrow \emptyset$ $\triangleright$ subset of S we've built up so far
  for $r = 1, \ldots, R$ **do** $\triangleright$ each iteration tries to recover 1 elt via M
    $T_r \leftarrow T_{r-1}$
    if $b_r = 1$ **then** $\triangleright$ this means Bob succeeds in round r
      $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{T_{r-1}})$ $\triangleright$ Invariant: $T_r = S \backslash S_r$
      $A \leftarrow A \cup \{s_r\},\ T_r \leftarrow T_r \cup \{s_r\}$
    end if
    Insert $m - n_r - |T_r|$ items from $B \backslash T_r$ into T_r with smallest π_i
    $\triangleright$ **"Differential Privacy"** step. Still n_r elements left to recover.
  end for
  return $B \cup A$
end procedure

Optimal lower bound for $\mathbf{R}_{\delta}^{\to,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
    $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
    $A \leftarrow \emptyset$ ▷ the set D recovers just from M

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
    $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
    $A \leftarrow \emptyset$ ▷ the set D recovers just from M
    $S_0 \leftarrow S$ ▷ at end of round r, D still needs to recover S_r

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
  $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
  $A \leftarrow \emptyset$ ▷ the set D recovers just from M
  $S_0 \leftarrow S$ ▷ at end of round r, D still needs to recover S_r
  for $r = 1, \ldots, R$ **do**
    $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ ▷ $s_r \overset{?}{\in} S_{r-1}$ found in round r

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$

  $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$

  $A \leftarrow \emptyset$ ▷ the set D recovers just from M

  $S_0 \leftarrow S$ ▷ at end of round r, D still needs to recover S_r

  for $r = 1, \ldots, R$ **do**

    $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ ▷ $s_r \overset{?}{\in} S_{r-1}$ found in round r

    $S_r \leftarrow S_{r-1}$

    if $s_r \in S_{r-1}$ **then** ▷ i.e. if s_r is a valid sample

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
  $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
  $A \leftarrow \emptyset$ ▷ the set D recovers just from M
  $S_0 \leftarrow S$ ▷ at end of round r, D still needs to recover S_r
  for $r = 1, \ldots, R$ **do**
    $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ ▷ $s_r \overset{?}{\in} S_{r-1}$ found in round r
    $S_r \leftarrow S_{r-1}$
    if $s_r \in S_{r-1}$ **then** ▷ i.e. if s_r is a valid sample
      $b_r \leftarrow 1$ ▷ $b \in \{0,1\}^R$ indicating which rounds succeed
      $A \leftarrow A \cup \{s_r\}$, $S_r \leftarrow S_r \setminus \{s_r\}$

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow, pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
  $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
  $A \leftarrow \emptyset$ ▷ the set D recovers just from M
  $S_0 \leftarrow S$ ▷ at end of round r, D still needs to recover S_r
  for $r = 1, \ldots, R$ **do**
    $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ ▷ $s_r \overset{?}{\in} S_{r-1}$ found in round r
    $S_r \leftarrow S_{r-1}$
    if $s_r \in S_{r-1}$ **then** ▷ i.e. if s_r is a valid sample
      $b_r \leftarrow 1$ ▷ $b \in \{0,1\}^R$ indicating which rounds succeed
      $A \leftarrow A \cup \{s_r\}$, $S_r \leftarrow S_r \setminus \{s_r\}$
    else
      $b_r \leftarrow 0$
    end if

Optimal lower bound for $\mathbf{R}_{\delta}^{\rightarrow,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

procedure $E(S)$
  $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
  $A \leftarrow \emptyset$ $\triangleright$ the set D recovers just from M
  $S_0 \leftarrow S$ $\triangleright$ at end of round r, D still needs to recover S_r
  for $r = 1, \ldots, R$ **do**
    $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ $\triangleright$ $s_r \stackrel{?}{\in} S_{r-1}$ found in round r
    $S_r \leftarrow S_{r-1}$
    if $s_r \in S_{r-1}$ **then** $\triangleright$ i.e. if s_r is a valid sample
      $b_r \leftarrow 1$ $\triangleright$ $b \in \{0,1\}^R$ indicating which rounds succeed
      $A \leftarrow A \cup \{s_r\}$, $S_r \leftarrow S_r \backslash \{s_r\}$
    else
      $b_r \leftarrow 0$
    end if
    remove $|S_r| - n_r$ elts from S_r with smallest π_i $\triangleright$ now $|S_r| = n_r$
  end for

Optimal lower bound for $\mathbf{R}_\delta^{\to,pub}(\mathbf{UR}^{\subset,+})$

Encoding algorithm for $S \subset [n]$, $|S| = m$

---

procedure $E(S)$
    $M \leftarrow \mathrm{Alice}(\mathbf{1}_S)$
    $A \leftarrow \emptyset$ $\quad\triangleright$ the set D recovers just from M
    $S_0 \leftarrow S$ $\quad\triangleright$ at end of round r, D still needs to recover S_r
    for $r = 1, \ldots, R$ **do**
        $s_r \leftarrow \mathrm{Bob}(M, \mathbf{1}_{S \setminus S_{r-1}})$ $\quad\triangleright$ $s_r \overset{?}{\in} S_{r-1}$ found in round r
        $S_r \leftarrow S_{r-1}$
        if $s_r \in S_{r-1}$ **then** $\quad\triangleright$ i.e. if s_r is a valid sample
            $b_r \leftarrow 1$ $\quad\triangleright$ $b \in \{0,1\}^R$ indicating which rounds succeed
            $A \leftarrow A \cup \{s_r\}$, $S_r \leftarrow S_r \setminus \{s_r\}$
        else
            $b_r \leftarrow 0$
        end if
        remove $|S_r| - n_r$ elts from S_r with smallest π_i $\triangleright$ now $|S_r| = n_r$
    end for
    return $(M,\ S \setminus A,\ b)$
end procedure

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$
- $|S_r| = n_r$ and $|S| = m$, so $H(S_r) \leq \log \binom{m}{n_r}$

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$
- $|S_r| = n_r$ and $|S| = m$, so $H(S_r) \leq \log \binom{m}{n_r}$
- We show that for *any* $T \in \binom{S}{n_r}$ and x,
  $\mathbb{P}(S_r = T | X = x) \leq p = \frac{2^{6K}}{\binom{m}{n_r}}$

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$
- $|S_r| = n_r$ and $|S| = m$, so $H(S_r) \leq \log \binom{m}{n_r}$
- We show that for *any* $T \in \binom{S}{n_r}$ and x,
  $\mathbb{P}(S_r = T | X = x) \leq p = \frac{2^{6K}}{\binom{m}{n_r}}$
  $\implies H(S_r|X) \geq \log \frac{1}{p} \geq \log \binom{m}{n_r} - 6K$ $\square$

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$
- $|S_r| = n_r$ and $|S| = m$, so $H(S_r) \leq \log \binom{m}{n_r}$
- We show that for *any* $T \in \binom{S}{n_r}$ and x,
  $\mathbb{P}(S_r = T | X = x) \leq p = \frac{2^{6K}}{\binom{m}{n_r}}$
  $\implies H(S_r|X) \geq \log \frac{1}{p} \geq \log \binom{m}{n_r} - 6K$ $\square$

Correctness of protocol then follows by adaptivity lemma.

Analysis

Recall $K = \lfloor \frac{1}{16} \log \frac{1}{\delta} \rfloor$. Note $n_r = 2^{-r/K} m \approx (1 - 1/K)^r m$.
X is randomness used by $\mathbf{UR}^{\subset,+}$ protocol.

Lemma: If in each round we add a random $1/K$-fraction of the remaining elements of S to T_r, then for all $r \in [R]$, $I(X; S_r) \leq 6K$.
Proof:

- $I(X; S_r) = H(S_r) - H(S_r|X)$
- $|S_r| = n_r$ and $|S| = m$, so $H(S_r) \leq \log \binom{m}{n_r}$
- We show that for *any* $T \in \binom{S}{n_r}$ and x,
  $\mathbb{P}(S_r = T | X = x) \leq p = \frac{2^{6K}}{\binom{m}{n_r}}$
  $\implies H(S_r|X) \geq \log \frac{1}{p} \geq \log \binom{m}{n_r} - 6K$ $\square$

Correctness of protocol then follows by adaptivity lemma.

Note a "$1/K$-fraction of what's left" requires at least K items left.
Thus we stop when $2^{-R/K} m < K$, i.e. $R = \Theta(K \log(m/K))$.

The End