A Note on Space Lower Bound for Samplers

Jelani Nelson, Jakub Pachocki, Zhengyu Wang

January 25, 2017

We study the space lower bound for maintaining a sampler over a turnstile stream. An ℓ_p-sampler with failure probability at most δ is a randomized data structure for maintaining vector $x \in \mathbb{R}^n$ (initially 0) under a stream of updates in the form of (i, Δ) (meaning that $x_i \leftarrow x_i + \Delta$); in the end, with probability at least $1 - \delta$, it gives an "ℓ_p-sample" according to x: namely, item i is sampled with probability $\frac{|x_i|^p}{\sum_{j \in [n]} |x_j|^p}$.

Note that updates are independent of the randomness used in the sampler. That is, for the purpose of proving a lower bound, we assume an oblivious adversary.

To the best of my knowledge, the best space upper bound for ℓ_0 sampler is $O(\log^2 n \log \frac{1}{\delta})$ bits, while the previous best lower bound is $\Omega(\log^2 n + \log \frac{1}{\delta})$ bits (where $\Omega(\log^2 n)$ is shown in [JST11]). The bound is tight for constant δ, while for example, when $\delta = \frac{1}{n}$, the gap is $\log n$.

We assume that

$$2^{-n^{c_1}} < \delta < (\log n)^{-c_2}, \tag{1}$$

where $c_1 = 0.01$ and $c_2 = 2$. For other range of δ we will study later.

In this note, we show space lower bounds for maintaining a sampler for a *binary* vector. That is, at any time, we are guaranteed that $x \in \{0, 1\}^n$. This makes our result strong in the sense that (1) the lower bound applies for any p; (2) the lower bound also works for strict turnstile streams.

In the following sections, we give sequentially improved lower bounds. First, we give a lower bound of $\Omega(\log n \log \frac{1}{\delta})$ bits. Then, we improve it to $\Omega(\frac{\log^2 n \log \frac{1}{\delta}}{(\log \log n + \log \log \frac{1}{\delta})^2})$ bits. The lower bounds are based on communication complexity in the public random coin model. Alice wants to send Bob a uniform random set $A \subseteq [n]$ of size m (Bob knows m, but the random source generating A is independent of the random source accessible to Bob). The one-way communication problem is: Alice sends some message to Bob, and Bob is required to recover A completely. Since the randomness in A contains $\log\binom{n}{m}$ bits of information, any randomized protocol that works with probability 1 requires at least $\log\binom{n}{m}$ expected bits.

Now Alice considers to attach (the memory of) a sampler SAMP in the message. The sampler uses public random coins as its random source, so that the sampler will behave the same at Alice's and Bob's as long as the updates are all the same. Alice will insert all the items in A into SAMP and send SAMP to Bob. In addition, Alice will send a subset $B \subseteq A$ to Bob, so that together with B and SAMP, Bob is able to recover A with good probability based on some protocol they have agreed on.

Now we turn the previous protocol into a new one without any failure. Let SUCC denote the event (or a subset of the event) that Bob successfully recovers A (note that Alice can simulate Bob, so she knows exactly when SUCC happens). If SUCC happens Alice will send Bob a message starting with a 1, followed by (the memory of) SAMP, then followed by the native encoding (explained later) of B; otherwise, Alice will send a message starting with a 0, followed by the native encoding of A. We say the native encoding of a set $S \subseteq [n]$ to be an integer (expressed in binary) in $[\binom{n}{|S|}]$ together with $|S|$ (taking $\log n$ bits). We drop the size of the set if it is known by the receiver.

Lemma 1. *Let s denote the space (in bits) used by a sampler with failure probability at most δ. Let s' denote the expected number of bits to represent B conditioned on SUCC (if we need to send some extra auxiliary information, we will also count it into s'). We have*

$$(1 + \mathsf{s} + \mathsf{s}') \cdot \mathbb{P}(\mathit{SUCC}) + (1 + \log\tbinom{n}{m}) \cdot (1 - \mathbb{P}(\mathit{SUCC})) \geq \log\tbinom{n}{m}.$$

If $\mathbb{P}(\mathit{SUCC}) \geq 1/2$, we have

$$\mathsf{s} \geq \log\tbinom{n}{m} - \mathsf{s}' - 2. \tag{2}$$

Remark 1. *Because the space lower bound in this note is proven via communication complexity under public random coin model, it also applies to non-uniform models of computation such as circuits and branching programs.*

Remark 2. *The space lower bound in this note still applies if the sampler is required to output an arbitrary item whose coordinate is non-zero instead of a uniformly random one.*

1 $\Omega(\log n \log \frac{1}{\delta})$ Bits Lower Bound

Let $m = \frac{1}{2}\log\frac{1}{\delta}$, namely, Alice wants to send a uniform random set $A \subseteq [n]$ of size $\frac{1}{2}\log\frac{1}{\delta}$ to Bob. Let $A = \{a_1, \ldots, a_m\}$ and $a_1 < \ldots < a_m$.

Algorithm 1 Alice's Encoder.

procedure $\mathrm{ENC}_1(A)$
    $\mathsf{SAMP} \leftarrow \emptyset$
    for $i = 1, 2, \ldots, m$ **do**
        Insert a_i into SAMP
    end for
    return SAMP
end procedure

Algorithm 2 Bob's Decoder.

procedure $\mathrm{DEC}_1(\mathsf{SAMP})$
    $S \leftarrow \emptyset$
    for $i = 1, 2, \ldots, m$ **do**
        Let SAMP_i be a copy of SAMP ▷ So that SAMP_i can behave as if it is SAMP
        for $s \in S$ **do** ▷ Enumerate the elements in S from smallest to biggest
            Remove s from SAMP_i
        end for
        Obtain a sample s_i from SAMP_i
        $S \leftarrow S \cup \{s_i\}$
    end for
    return S
end procedure

Lemma 2. *For any $A \subseteq [n]$, where $|A| = m = \frac{1}{2}\log\frac{1}{\delta}$, $\mathbb{P}(\mathit{DEC}_1(\mathit{ENC}_1(A)) = A) \geq 1/2$.*

Proof. Let E_S denote the event that after removing all the items in S (in the order from smallest to biggest) from SAMP, it gives a valid sample when queried. We have

$$\mathbb{P}(\mathsf{DEC}_1(\mathsf{ENC}_1(A)) = A) \geq \mathbb{P}(\bigcap_{S \subseteq A, S \neq \emptyset} E_S) \geq 1 - \sum_{S \subseteq A, S \neq \emptyset} \mathbb{P}(\overline{E_S}) \geq 1 - \delta \cdot 2^{\frac{1}{2}\log\frac{1}{\delta}} \geq 1/2.$$

$\square$

Lemma 3. $\mathsf{s} = \Omega(\log n \log \frac{1}{\delta})$ *for* $2^{-n^{0.99}} < \delta < \frac{1}{4}$.

Proof. It follows from Formula 2 in Lemma 1, where $\log \binom{n}{\frac{1}{2}\log\frac{1}{\delta}} = \Omega(\log n \log \frac{1}{\delta})$ and $\mathsf{s}' = 0$. $\square$

Remark 3. *The following decoder DEC_1' is similar to DEC_1, but we will lose a factor of $\log\log\frac{1}{\delta}$ in the lower bound because by doing so we have to union-bound $O(m!)$ events instead of $O(2^m)$ events (so that in turn we have to set m to be $\frac{\log\frac{1}{\delta}}{\log\log\frac{1}{\delta}}$ in order to have good success probability).*

Algorithm 3 A Worse Decoder.

procedure $\mathrm{DEC}_1'(\mathsf{SAMP})$
    $S \leftarrow \emptyset$
    for $i = 1, 2, \ldots, m$ **do**
        Obtain a sample s_i from SAMP
        Remove s_i from SAMP
        $S \leftarrow S \cup \{s_i\}$
    end for
    return S
end procedure

2 $\Omega(\frac{\log^2 n \log\frac{1}{\delta}}{(\log\log n + \log\log\frac{1}{\delta})^2})$ Bits Lower Bound

In the previous section we have shown how to extract $\Theta(\log\frac{1}{\delta})$ words of information from a sampler. Our goal is to extract more words. New observation comes from the upper bound for constructing an ℓ_0 sampler. When $\delta = \frac{1}{n}$, we have the following sampler algorithm that consumes $O(\log^3 n)$ bits. The sampler consists of $\log n$ layers, and on layer $i \in [\log n]$ it maintains a separate $\log n$-sparse recover system for the sub-stream generated by sub-sampling the items from the universe $[n]$ with probability 2^{-i}. Each sparse recovery system takes $O(\log^2 n)$ bits. Its correctness comes from the fact that

1. With probability at least $1 - n^{-c}$ there is some layer i that contains at least 1 and at most $\log n$ items whose coordinates are non-zero.

2. Conditioned on the event that on layer i the number of items whose coordinates are non-zero is between 1 and $\log n$, the sparse recovery system on layer i works with failure probability at most n^{-c}. In this context, we say the sparse recovery system works if it could recover at least one item.

Intuitively, the previous lower bound only extracts information from one single layer (i.e. layer 0). In this section, we build a framework to extract information from multiple layers. A second technique that makes our improvement possible is bundling: after obtaining a sample from the sampler, we not only peel off the sample, but also remove the whole bundle containing the sample from the sampler, where bundles are a partition of elements inserted to the sampler.

2.1 Protocol

Alice wants to send random set A to Bob where $|A| = m$. Similar to ENC_1, Alice constructs SAMP by inserting all the items in A, and sends it to Bob. Moreover, Alice will send Bob a subset $B \subseteq A$ computed as follows. Initially $B = A$. Alice proceeds in R rounds. On round r ($r = 1, \ldots, R$) Alice considers interval I_r (where $I_1 = \{0, \ldots, n-1\}$), and divides I_r into K even parts: $I_{r,1}, \ldots, I_{r,K}$. She samples a uniform random k_r in $1, \ldots, K$, and sets the next interval to be $I_{r+1} = I_{r,k_r}$. She removes all elements except $A \cap I_r$ from SAMP, denoted by SAMP_r, and she wants to obtain n_r items in $A \cap I_r$ from SAMP_r: each time she obtains a sample s, she finds the k in which $s \in I_{r,k}$, and remove $A \cap I_{r,k}$ from SAMP_r. If s is an invalid sample or $k = k_r$, Alice's encoder fails; otherwise she remove s from B, and s is considered to be the save with the help of SAMP. The decoding process is symmetric.

The parameters, encoder and decoder are given as follows.

Algorithm 4 Variables Shared by Alice's ENC_4 and Bob's DEC_4.

$m \leftarrow n^{0.99}$
$K \leftarrow \log n \cdot \log \frac{1}{\delta}$
$R \leftarrow \frac{\log n}{50 \log K}$
$i_1 \leftarrow 0$
$\Delta_1 \leftarrow n$
for $r = 1, \ldots, R-1$ **do**
Let k_r be a uniform sample from $\{1, \ldots, K\}$
$i_{r+1} \leftarrow i_r + \frac{k_r - 1}{K} \Delta_r$
$\Delta_{r+1} \leftarrow \frac{\Delta_r}{K}$
end for
for $r = 1, \ldots, R$ **do**
$n_r \leftarrow \frac{1}{2} \cdot \frac{\log \frac{1}{\delta}}{\log K}$
$I_r \leftarrow \{i \in \mathbb{N} | i_r \le i < i_r + \Delta_r\}$
for $k = 1, \ldots, K$ **do**
$I_{r,k} \leftarrow \{i \in \mathbb{N} | i_r + (k-1)\frac{\Delta_r}{K} \le i < i_r + k\frac{\Delta_r}{K}\}$
end for
end for

Algorithm 5 Alice's Encoder.

procedure $\mathrm{ENC}_4(A)$
$\mathsf{SAMP} \leftarrow \emptyset$
Insert all elements in A into SAMP
$B \leftarrow A$
for $r = 1, \ldots, R$ **do**
Let SAMP_r be a copy of SAMP
Remove all elements in $A \backslash (A \cap I_r)$ from SAMP_r
for $i = 1, \ldots, n_r$ **do**
Obtain a sample s from SAMP_r
if s is not valid **then**
return "**fail**"
end if
Find k such that $s \in I_{r,k}$
if $k = k_r$ **then**
return "**fail**"
end if
Remove all elements in $A \cap I_{r,k}$ from SAMP_r
$B \leftarrow B \backslash \{s\}$
end for
end for
return (SAMP, B)
end procedure

Algorithm 6 Bob's Decoder.

procedure $\text{DEC}_4(\mathsf{SAMP}, B)$
 $A \leftarrow B$
 for $r = 1, 2, \ldots, R$ **do**
  Let SAMP_r be a copy of SAMP
  Remove all the items in $A \backslash (A \cap I_r)$ from SAMP_r
  for $i = 1, \ldots, n_r$ **do**
   Obtain a sample s from SAMP_r
   $A \leftarrow A \cup \{s\}$
   Find k such that $k \in I_{r,k}$
   Remove items in $A \cap I_{r,k}$ from SAMP_r
  end for
 end for
 return A
end procedure

2.2 Analysis

Note that $\mathbb{E}(|A \cap I_{r,k}|) = m \cdot K^{-r}$. By the choice of parameters we have $m = n^{0.99}$, $K^{-r} \geq K^{-R} = n^{-0.02}$ (note that the range of δ is specified by Formula 1). Therefore $\mathbb{E}(|A \cap I_{r,k}|) \geq n^{0.97}$, for $r = 1, \ldots, R$ and $k = 1, \ldots, K$. For any pair of (r, k), because of the randomness in A, we can prove the probability that $A \cap I_{r,k}$ is empty is exponentially small. By union bound, with high probability, all $A \cap I_{r,k}$ are not empty. In the following, we will discuss conditioned on that.

Lemma 4. *With probability at least $\frac{9}{10}$, ENC_4 does not return "fail".*

Proof. The probability that ENC_4 returns fail on line 11 is at most $\sum_{i=1}^{R} K^{n_i} \cdot \delta = \frac{\delta^{1/2} \log n}{50 \log K} < \frac{1}{50}$.

The probability that ENC_4 returns fail on line 15 is at most $\sum_{r=1}^{R} \frac{n_r}{K} = \frac{1}{100 \log^2 K} < \frac{1}{100}$. □

Lemma 5. *If ENC_4 does not return "fail", we have $\mathsf{DEC}_4(\mathsf{ENC}_4(A)) = A$.*

Lemma 6. *If ENC_4 does not return "fail", we have $|A| - |B| = \Omega(\frac{\log n \log \frac{1}{\delta}}{(\log\log n + \log\log \frac{1}{\delta})^2})$, where B is the set ENC_4 outputs.*

Proof. $|A| - |B| = \sum_{r=1}^{R} n_r = \frac{\log n}{50 \log K} \cdot \frac{1}{2} \cdot \frac{\log \frac{1}{\delta}}{\log K} = \Omega(\frac{\log n \log \frac{1}{\delta}}{(\log\log n + \log\log \frac{1}{\delta})^2})$. □

Lemma 7. *Let $m = n^{0.99}$. Let $X \in \mathbb{N}$ be a random variable, and $X \leq m$. Moreover, $\mathbb{E}(X) \leq m - d$. We have $\mathbb{E}(\log \binom{n}{m} - \log \binom{n}{X}) = \Omega(d \log n)$.*

Proof.

$$
\begin{aligned}
\log \binom{n}{m} - \log \binom{n}{X} &= \log \frac{n!/(m!(n-m)!)}{n!/(X!(n-X)!)} \\
&= \sum_{i=1}^{m-X} \log \frac{n - X - i + 1}{m - i + 1} \\
&\geq (m - X) \cdot \log \frac{n - X}{m} \\
&\geq (m - X) \cdot \log n^{1/200}
\end{aligned}
$$

Taking expectation on both sides, we get $\mathbb{E}(\log \binom{n}{m} - \log \binom{n}{X}) \geq \frac{d}{200} \log n$. □

Theorem 1. $\mathsf{s} = \Omega(\frac{\log^2 n \log \frac{1}{\delta}}{(\log\log n + \log\log \frac{1}{\delta})^2})$ *for* $2^{-n^{0.01}} < \delta < (\log n)^{-2}$.

Proof. Let SUCC be the conjunction of the following events:

1. All $A \cap I_{r,k}$ are non-empty ($r = 1, \ldots, R, k = 1, \ldots, K$).
2. ENC_4 does not return "fail" on input A.

By Lemma 4, we have $\mathbb{P}(\mathsf{SUCC}) \geq \frac{1}{2}$. By Lemma 1, we have $\mathsf{s} \geq \log\binom{n}{m} - s' - 2$. By definition, $s' = \log n + \mathbb{E}(\log\binom{n}{|B|} | \mathsf{SUCC})$. By Lemma 6 and Lemma 7, we get $\mathbb{E}(\log\binom{n}{|B|} | \mathsf{SUCC}) = \log\binom{n}{m} - \Omega(\frac{\log^2 n \log \frac{1}{\delta}}{(\log\log n + \log\log \frac{1}{\delta})^2})$. □

3 $\Omega(\log^2 n \log \frac{1}{\delta})$ Bits Lower Bound

Let $R = \frac{1}{10} \log n \log \frac{1}{\delta}$ and let $K = \log \frac{1}{\delta}$.

Algorithm 7 Alice's Encoder.

procedure ENC(A)
  $\mathsf{SAMP} \leftarrow \emptyset$
  Insert items in A into SAMP
  $A_1 \leftarrow A$
  $S \leftarrow \emptyset$
  for $r = 1, \ldots, R$ **do**
    Let SAMP_r be a copy of SAMP
    Remove items in $A \backslash A_r$ from SAMP_r ▷ Now SAMP_r contains the elements in A_r
    Let s_r be a sample from SAMP_r
    $S \leftarrow S \cup \{s_r\}$
    $A_{r+1} \leftarrow A_r \backslash \{s_r\}$
    for $a \in A_{r+1}$ **do**
      With probability $\frac{1}{K}$, $A_{r+1} \leftarrow A_{r+1} \backslash \{a\}$ ▷ Use the randomness that Alice and Bob share
    end for
  end for
  return ($A \backslash S$, SAMP)
end procedure

Algorithm 8 Bob's Decoder.

procedure DEC(B, SAMP)
  $S \leftarrow \emptyset$
  $C_1 \leftarrow \emptyset$
  for $r = 1, \ldots, R$ **do**
    Let SAMP_r be a copy of SAMP
    Remove items in C_r from SAMP_r
    Let s_r be a sample from SAMP_r
    $S \leftarrow S \cup \{s_r\}$
    $C_{r+1} \leftarrow C_r \cup \{s_r\}$
    for $a \in B \backslash C_{r+1}$ **do**
      With probability $\frac{1}{K}$, $C_{r+1} \leftarrow C_{r+1} \cup \{a\}$ ▷ Use the randomness that Alice and Bob share,
    end for ▷ so that $C_r = A \backslash A_r$ (A, A_r are defined in ENC)
  end for
  return $B \cup S$
end procedure

3.1 Analysis

Lemma 8. *Let function $f\colon \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$. Let X be a uniformly random string in $\{0,1\}^n$. If for any $y \in \{0,1\}^m$ we have $\mathbb{P}(f(X,y)=1) \le \delta$ where $0 < \delta < 1$, then for any random variable Y in $\{0,1\}^m$, we have*

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

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

Proof. It is equivalent to prove $I(X;Y) \ge \mathbb{E}(f(X,Y)) \cdot \log \frac{1}{\delta} - 1$. By definition of mutual entropy, $I(X;Y) = H(X) - H(X|Y)$ where $H(X) = n$ and $H(X|Y) \le 1 + (1 - \mathbb{E}(f(X,Y))) \cdot n + \mathbb{E}(f(X,Y)) \cdot (n - \log \frac{1}{\delta}) = n + 1 - \mathbb{E}(f(X,Y)) \cdot \log \frac{1}{\delta}$. The upper bound for $H(X|Y)$ is obtained by considering the following one-way communication problem: Alice obtains both X and Y while Bob only gets Y, what is the (minimum) expected number of bits that Alice sends to Bob so that Bob can recover X? Any protocol gives an upper bound for $H(X|Y)$, and we simply take the following protocol: first Alice sends Bob $f(X,Y)$ (taking 1 bit); and then if $f(X,Y)=0$ Alice sends X directly (taking n bits), otherwise, $f(X,Y)=1$, Alice sends the index of X in $\{x|f(x,Y)=1\}$ (taking $\log(\delta 2^n) = n - \log \frac{1}{\delta}$ bits). □

We do the analysis conditioned on A. Let X denote the random source used by the sampler. The probability that the sampler fails is upper bounded by $\sum_{r=1}^{R} 2^{I(X;A_r)} \cdot \delta$.

References

[JST11] Hossein Jowhari, Mert Sağlam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 49–58. ACM, 2011.