Lower bounds on frequency estimation of data streams

Sumit Ganguly*
Indian Institute of Technology, Kanpur

Abstract. We consider a basic problem in the general data streaming model, namely, to estimate a vector $f \in \mathbb{Z}^n$ that is arbitrarily updated (i.e., incremented or decremented) coordinate-wise. The estimate $\hat{f} \in \mathbb{Z}^n$ must satisfy $\|\hat{f} - f\|_\infty \leq \epsilon \|f\|_1$, that is, $\forall i (|\hat{f}_i - f_i| \leq \epsilon \|f\|_1)$. It is known to have $O(\epsilon^{-1})$ randomized space upper bound [4], $\Omega(\epsilon^{-1} \log(\epsilon n))$ space lower bound [2] and deterministic space upper bound of $\tilde{O}(\epsilon^{-2})$ bits [2]. We show that any deterministic algorithm for this problem requires space $\Omega(\epsilon^{-2} \log\|f\|_1)$ bits.

1 Introduction

A data stream $\sigma$ over the domain $[1, n] = \{1, 2, \ldots, n\}$ is modeled as a sequence of records of the form $(pos, i, \delta v)$, where, $pos$ is the current sequence index, $i \in [1, n]$ and $\delta v \in \{+1, -1\}$. Here, $\delta v = 1$ signifies an insertion of an instance of $i$ and $\delta v = -1$ signifies a deletion of an instance of $i$. For each data item $i \in [1, n]$, its frequency ($freq$ $\sigma$), is defined as $\sum_{(pos, i, \delta v) \in \text{stream}} \delta v$. The size of $\sigma$ is defined as $|\sigma| = \max\{|freq \sigma'|_\infty \mid \sigma'$ prefix of $\sigma\}$. In this paper, we consider the general stream model, where, the $n$-dimensional frequency vector $freq \sigma \in \mathbb{Z}^n$. The data stream model of processing permits online computations over the input sequence using sub-linear space. The data stream computation model has proved to be a viable model for a number of application areas, such as network monitoring, databases, financial data processing, etc..

We consider the problem APPROXFREQ($\epsilon$): given a data stream $\sigma$, return $\hat{f}$, such that $err(\hat{f}, freq \sigma) \leq \epsilon$, where, the function $err$ is given by (1). Equivalently, the problem may be formulated as: given $i \in [1, n]$, return $\hat{f}_i$ such that $|\hat{f}_i - (freq \sigma)_i| \leq \epsilon \cdot |freq \sigma|_1$, where, $\|f\|_1 = \sum_{i \in [1, n]} |f_i|$.

$$err(\hat{f}, f) \overset{\text{def}}{=} \frac{\|\hat{f} - f\|_\infty}{\|f\|_1} \leq \epsilon \qquad (1)$$

The problem APPROXFREQ($\epsilon$) is of fundamental interest in data streaming applications. For general streams, this problem is known to have a space lower bound of $\Omega(\epsilon^{-1} \log(n))$ [2], a randomized space upper bound of $\tilde{O}(\epsilon^{-1})$ [4], and a deterministic space upper bound of $\tilde{O}(\epsilon^{-2})$ bits [7]. For insert-only streams (i.e., $freq \sigma \geq 0$), there exist deterministic algorithms that use $O((\epsilon^{-1} \log(mn)))$ space [5][7]; however extensions of these algorithms to handle deletions in the stream are not known.

Mergeability. Data summary structures for summarizing data streams for frequency dependent computations (e.g., approximate frequent items, frequency moments, etc.; formally defined in Section 2) typically exhibit the property of arbitrary mergeability. If $D$ is a data structure for processing a stream and $D_j$, $j = 1, \ldots, k$ for $k$ arbitrary, be the respective current state of the structure after processing streams $S_j$, then, there exists a simple operation Merge such that $\text{Merge}(D_1, \ldots, D_k)$ reconstructs the state of $D$ that would be obtained by
processing the union of streams \( S_j, j = 1, 2, \ldots, k \). For randomized summaries, this might require initial random seeds to be shared. Thus, a summary of a distributed stream can be constructed from the summaries of the individual streams, followed by the Merge operation. Almost all known data streaming structures are arbitrarily mergeable, including, sketches \([1]\), COUNTSKETCH \([3]\), COUNT-MIN sketches \([4]\), Flajolet-Martin sketches \([6]\) and its variants, \( k \)-set \([8]\), \( \text{CR-precis} \) structure \([7]\) and random subset sum \([10]\). In this paper, we ask the question, namely, when are stream summaries mergeable?

**Contributions.** We present a space lower bound of \( \Omega(\epsilon^{-2}(\log m)) - O(\log n) \) bits for any deterministic uniform algorithm \( A_n \) for the problem APPROXFREQ(\( \epsilon \)) over input streams of size \( m \) over the domain \([1, n]\), where, \( 1/(24\sqrt{n}) \leq \epsilon \leq 1/32 \). The uniformity is in the sense that \( A_n \) must be able to solve APPROXFREQ(\( \epsilon \)) for all general input streams over the domain \([1, n]\). The lower bound implies that the \( \text{CR-precis} \) structure \([7]\) is nearly space-optimal for APPROXFREQ(\( \epsilon \)), up to poly-logarithmic factors. The uniformity requirement is essential since there exists an algorithm that solves APPROXFREQ(\( \epsilon \)) for all input streams \( \sigma \) with \( |\sigma| \leq 1 \) using space \( O(\epsilon^{-1}\text{polylog}(n)) \) \([9]\).

We also show that for any deterministic and uniform algorithm \( A_n \) over general streams, there exists another algorithm \( B_n \) such that (a) the state of \( B_n \) is arbitrarily mergeable, (b) \( B_n \) uses at most \( O(\log n) \) bits of extra space than \( A_n \), and, (c) for every input stream \( \sigma \), the output of \( B_n \) on \( \sigma \) is the same as the output of \( A_n \) on some stream \( \sigma' \) such that \( \text{freq} \sigma = \text{freq} \sigma' \). In other words, if \( A_n \) correctly solves a given frequency dependent problem, so does \( B_n \); further, the state of \( B_n \) is arbitrarily mergeable and \( B_n \) uses \( O(\log n) \) bits of extra space. This shows that deterministic data stream summaries for frequency dependent computation are essentially arbitrarily mergeable.

## 2 Stream Automaton

In this section, we define a stream automaton and study some basic properties.

**Definition 1 (Stream Automaton).** A stream automaton \( A_n \) over the domain \([1, n]\) is a deterministic Turing machine that uses two tapes, namely, a two-way read-write work-tape and a one-way read-only input tape. The input tape contains the input stream \( \sigma \). After processing its input, the automaton writes an output, denoted by \( \text{output}_{A_n}(\sigma) \), on the work-tape.

Effective space usage. We say that a stream automaton uses space \( s(n, m) \) bits if for all input streams \( \sigma \) having \( |\sigma| \leq m \), the number of cells (bits) on the work-tape in use, after having processed \( \sigma \), is bounded by \( s(n, m) \). In particular, this implies that for \( m \geq m' \), \( s(n, m) \geq s(n, m') \). The space function \( s(n, m) \) does not count the space required to actually write the answer on the work-tape, or to process the \( s(n, m) \) bits of the work-tape once the end of the input tape is observed. The proposed model of stream automata is non-uniform over the domain size \( n \), (and uniform over the stream size parameter \( m = |\sigma| \)), since, for each \( n \geq 1 \), there is a stream automaton \( A_n \) for solving instances of a problem over domain size \( n \). This creates a problem in quantifying effective space usage, particularly, for low-space computations, that is, \( s(n, m) = o(n \log m) \). Let \( Q(A_n) \) denote the set of states in the finite control of the automaton \( A_n \). If \( |Q(A_n)| \geq m2^n \), then, for all \( m' \leq m \), the automaton can map the frequency vector isomorphically into its finite control, and \( s(n, m) = 0 \). This problem is caused by non-uniformity of the model as a function of the domain size \( n \), and can be avoided as follows. We define the effective space usage of \( A_n \) as

\[
\text{Space}(A_n, m) \overset{\text{def}}{=} s(n, m) + \log s(n, m) + |Q(A_n)|.
\]
Although, the model of stream automata does not explicitly allow queries, this can be modeled by a stream automaton’s capability of writing vectors as answers, whose space is not counted towards the effective space usage. So if \( \{q_i\}_{i \in I} \) denotes the family of all queries that are applicable for the given problem, where, \( I \) is a finite index set of size \( p(n) \) then, the output of the automaton can be thought of as the \( p(n) \)-dimensional vector output\( A_n(\sigma) \).

A frequency dependent problem over a data stream is characterized by a family of binary predicates \( P_n(f, \text{freq } \sigma), f \in \mathbb{Z}^{p(n)}, n \geq 1 \), called the characteristic predicate for the domain \([1, n] \). \( P_n \) defines the acceptability (or good approximations) of the output. A stream automaton \( A_n \) solves a problem provided, for every stream \( \sigma \), \( P_n(\text{output } A_n(\sigma), \text{freq } \sigma) \) holds. For example, the characteristic predicate corresponding to the problem APPROXFREQ(\( \epsilon \)) is \( \text{err}(\hat{f}, f) \leq \epsilon \), where, \( \hat{f} \in \mathbb{Z}^n \) and \( \text{err}(\cdot, \cdot) \) is defined by (\( \square \)). Examples of frequency dependent problems are approximating frequencies and finding frequent items, approximate quantiles, histograms, estimating frequency moments, etc..

Given stream automata \( A_n \) and \( B_n \), \( B_n \) is said to be an output restriction of \( A_n \), provided, for every stream \( \sigma \), there exists a stream \( \sigma' \) such that, \( \text{freq } \sigma = \text{freq } \sigma' \) and output\( B_n(\sigma) = \text{output } A_n(\sigma') \). The motivation of this definition is the following straightforward lemma.

**Lemma 1.** Let \( P_n \) be the characteristic predicate of a frequency-dependent problem over data streams and suppose that a stream automaton \( A_n \) solves \( P_n \). If \( B_n \) is an output restriction of \( A_n \), then, \( B_n \) also solves \( P_n \). \( \square \)

**Proof.** Let \( \sigma \) be any input stream to \( B \) and let \( \hat{f} = \text{output } B(\sigma) \) be the output of \( B \) on \( \sigma \). Since, \( B \) is an output restriction of \( A \), hence, \( \hat{f} = \text{output } A(\sigma) \), for some stream \( \sigma \). Since, \( A \) solves \( P \), therefore, \( (\hat{f}, \text{freq } \sigma') \in P \). However, \( \text{freq } \sigma' = \text{freq } \sigma \), and therefore, \( (\hat{f}, \text{freq } \sigma) \in P \). Since, this holds for all \( \sigma \), \( B \) solves \( P \) as well. \( \square \)

**Notation.** Fix a value of the domain size \( n \geq 2 \). Each stream record of the form \((i, 1)\) and \((i, -1)\) is equivalently viewed as \(e_i\) and and \(-e_i\) respectively, where, \(e_i = [0, \ldots, 0, 1\ (\text{position } i), 0 \ldots, 0] \) is the \( i \)-th standard basis vector of \( \mathbb{R}^n \). A stream is thus viewed as a sequence of elementary vectors (or its inverse). The notation \( \sigma \circ \tau \) refers to the stream obtained by concatenating the stream \( \tau \) to the end of the stream \( \sigma \). In this notation, \( \text{freq } e_i = e_i \), \( \text{freq } -e_i = -e_i \) and \( \text{freq } \sigma \circ \tau = \text{freq } \sigma + \text{freq } \tau \). The inverse stream corresponding to \( \sigma \) is denoted as \( \sigma^r \) and is defined inductively as follows: \( e_i^r = -e_i, \ -e_i^r = e_i \) and and \( (\sigma \circ \tau)^r = \tau^r \circ \sigma^r \). The configuration of \( A_n \) is modeled as the triple \((q, h, w)\), where, \(q\) is the current state of the finite control of \( A_n \), \( h\) is the index of the current cell of the work tape, and \(w\) is the current contents of the work-tape. The processing of each record by \( A_n \) can be viewed as a transition function \( \oplus_{A_n}(a, v) \), where, \(a\) is the current configuration of \( A_n \), and \(v\) is the next stream record, that is, one of the \(e_i\)’s. The transition function is written in infix form as \(a \oplus_{A_n} v\). We assume that \( \oplus_{A_n} \) associates from the left, that is, \(a \oplus_{A_n} u_1 \circ u_2 \) means \((a \oplus_{A_n} u_1) \oplus_{A_n} u_2 \). Given a stream automaton \( A_n \), the space of possible configurations of \( A_n \) is denoted by \( C(A_n) \). Let \( C_m(A_n) \) denote the subset of configurations that are reachable from the initial state \( o \) and after processing an input stream \( \sigma \) with \( |\sigma| = ||\text{freq } \sigma||_{\infty} \leq m \). We now define two sub-classes of stream automata.

**Definition 2.** A stream automaton \( A_n \) is said to be **path independent**, if for each configuration \( s \) of \( A_n \) and input stream \( \sigma \), \(s \oplus_{A_n} \sigma\) is dependent only on \( \text{freq } \sigma \) and \( s \). A stream automaton \( A_n \) is said to be **path reversible** if for every stream \( \sigma \) and configuration \( s \), \( s \oplus_{A_n} \sigma \circ \sigma^r = s \), where, \( \sigma^r \) is the inverse stream of \( \sigma \). \( \square \)

**Overview of Proof.** The proof of the lower bound on the space complexity of APPROXFREQ(\( \epsilon \)) proceeds in three steps. A subclass of path independent stream automata, called **free au-
tomata is defined and is proved to be the class of path independent automata whose transition function $\oplus_{A_n}$ can be modeled as a linear mapping of $\mathbb{R}^n$, with input restricted to $\mathbb{Z}^n$. We then derive a space lower bound for APPRXFREQ($\epsilon$) for free automata (Section 4.1). In the second step, we show that a path independent automaton that solves APPRXFREQ($\epsilon$) can be used to design a free automaton that solves APPRXFREQ(4$\epsilon$)(Section 4.2). In the third step, we prove that for any frequency-dependent problem with characteristic predicate $P_n$ and a stream automaton $A_n$ that solves it, there exists an output-restricted stream automaton $B_n$ that also solves $P_n$, is path-independent, and, $\text{Space}(B_n, m) \leq \text{Space}(A_n, m) + O(\log n)$. This step has two parts—the property is first proved for the class of path-reversible automata $A_n$ (Section 5) and then generalized to all stream automata (Section 6). Combining the results of the three steps, we obtain the lower bound.

3 Path-independent stream automata

In this section, we study the properties of path independent automata. Let $A_n$ be a path-independent stream automaton over the domain $[1, n]$ and let $\oplus$ abbreviate $\oplus_{A_n}$. Define the function $+ : \mathbb{Z}^n \times C(A_n) \rightarrow C(A_n)$ as follows.

$$x + a = a \oplus \sigma \text{ where, } \text{freq } \sigma = x.$$ 

Since $A_n$ is a path independent automaton, the function $x + a$ is well-defined. The kernel $M_{A_n}$ of a path independent automaton is defined as follows. Let the initial configuration be denoted by $o$.

$$M_{A_n} = \{x \in \mathbb{Z}^n \mid x + o = 0 + o\}$$

The subscript $A_n$ in $M_{A_n}$ is dropped when $A_n$ is clear from the context.

**Lemma 2.** The kernel of a path independent automaton is a sub-module of $\mathbb{Z}^n$.

**Proof.** Let $x \in M$. Then, $0 + o = -x + x + o = -x + o$, or $-x \in M$. If $x, y \in M$, then, $0 + o = x + o = x + y + o$, or, $x + y \in M$. So $M$ is a sub-module of $\mathbb{Z}^n$. $\square$

The quotient set $\mathbb{Z}^n/M = \{x + M \mid x \in \mathbb{Z}^n\}$ together with the well-defined addition operation $(x + M) + (y + M) = (x + y) + M$, forms a module over $\mathbb{Z}$.

**Lemma 3.** Let $M$ be the kernel of a path independent automaton $A_n$. The mapping $x + M \mapsto x + o$ is a set isomorphism between $\mathbb{Z}^n/M$ and the set of reachable configurations $\{x + o \mid x \in \mathbb{Z}^n\}$. The automaton $A_n$ gives the same output for each $y \in x + M, x \in \mathbb{Z}^n$.

**Proof.** $y \in x + M$ iff $x - y \in M$ or $-y + x + o = o$, or, $x + o = y + o$. Thus, $A_n$ attains the same configuration after processing both $x$ and $y$ and therefore $A_n$ gives the same output for both $x$ and $y$. Since, $x + o = y + o$ iff $x - y \in M$, which implies that the mapping $x + M \mapsto x + o$ is an isomorphism. $\square$

Let $\mathbb{Z}^n_m$ denote the subset $\{-m, \ldots, m\}^n$ of $\mathbb{Z}^n$.

**Lemma 4.** Let $A_n$ be a path independent automaton with kernel $M$. Then,

$$\text{Space}(A_n, m) \geq \lceil \log|\{x + M \mid x \in \mathbb{Z}^n_m\}| \rceil \geq (n - \dim M) \log(2m + 1).$$
Proof. The set of distinct configurations of $A_n$ after it has processed a stream with frequency $x \in \mathbb{Z}^n_m$ is isomorphic to $\{x + M \mid x \in \mathbb{Z}^n_m\}$. The number of configurations using workspace of $s = s(n, m)$ is at most $|Q_{A_n}| \cdot s \cdot 2^s$. Therefore,

$$2^{\text{Space}(A_n, m)} = |Q_{A_n}| \cdot s \cdot 2^s \geq \left|\{x + M \mid x \in \mathbb{Z}^n_m\}\right| \cdot (|\mathbb{Z}^n_m|/(|M \cap \mathbb{Z}^n_m|) \geq (2m + 1)^n / (2m + 1)^{-r}.
$$

We now obtain an upper bound on the size $|M \cap \mathbb{Z}^n_m|$. Let $b_1, b_2, \ldots, b_r$ be a basis for $M$. The set

$$P_m = \{\alpha_1 b_1 + \ldots + \alpha_r b_r \mid |\alpha_i| \leq m \text{ and integral, } i = 1, 2, \ldots, n\}$$

defines the set of all integral points generated by $b_1, b_2, \ldots, b_r$ with multipliers in $\{-m, \ldots, m\}$. Thus,

$$|M \cap \mathbb{Z}^n_m| \leq |P_m| = (2m + 1)^r. \tag{3}$$

It follows that

$$\left|\{x + M \mid x \in \mathbb{Z}^n_m\}\right| \geq |\mathbb{Z}^n_m|/(|M \cap \mathbb{Z}^n_m| \geq (2m + 1)^n / (2m + 1)^{-r}.$$

Since, $r = \dim M$, substituting in (2) and taking logarithms, we have

$$\text{Space}(A_n, m) \geq \log \left|\{x + M \mid x \in \mathbb{Z}^n_m\}\right| \geq (n - r) \log(2m + 1). \quad \Box$$

Lemma 5 shows that given a sub-module $M$, a path-independent automaton with a given $M$ as a kernel can be constructed using nearly optimal space. The transition function $(x + M) + (y + M) = (x + y) + M$ implies that the state of a path independent automaton is arbitrarily mergeable.

**Lemma 5.** For any sub-module $M$ of $\mathbb{Z}^n$, one can construct a path-independent automaton with kernel $M$ that uses nearly optimal space $s(n, m) = \log \left|\{x + M \mid x \in \mathbb{Z}^n_m\}\right| + O(\log n)$ and uses $n^{O(1)}$ states in its finite control. \quad \square

Proof. Let $M$ be a given sub-module of $\mathbb{Z}^n$ with basis $b_1, \ldots, b_r$ (say). It is sufficient to construct a path independent automaton whose configurations are isomorphic to $E = \mathbb{Z}^n/M$. Since, $\mathbb{Z}^n$ is free, $\mathbb{Z}^n/M$ is finitely generated using any basis of $\mathbb{Z}^n$. Therefore, the basic module decomposition theorem states that

$$\mathbb{Z}^n/M = \mathbb{Z}/(q_1) \oplus \cdots \oplus \mathbb{Z}/(q_r).$$

where, $q_1|q_2|\cdots|q_r$. (Here, $\oplus$ refers to the direct sum of modules.) The finite control of the automaton stores $q_1, \ldots, q_r$ and the machinery required to calculate $1 \mod q_j$ and $-1 \mod q_j$ for each $j$. For the frequency vector $f$, the residue vector $f + M$ is maintained as a vector of residues with respect to the $q_j$’s as given by (4). Since, (4) is a direct sum, hence, the space used by this representation is optimal and equal to $\left|\{x + M \mid x \in [-m \ldots m]^n\}\right|$. \quad \Box

**Definition 3 (Free Automaton).** A path independent automaton $A_n$ with kernel $M$ is said to be free if $\mathbb{Z}^n/M$ is a free module. \quad \square

That is, $A_n$ is free if for every $x \in \mathbb{Z}^n$ such that there exists $a \in \mathbb{Z}$, $a \neq 0$ and $ax \in M$, it is the case that $x \in M$. For free automata $A_n$, it follows that $\mathbb{Z}^n$ is the direct sum of $M$ and $\mathbb{Z}^n/M$, that is, $\mathbb{Z}^n = \mathbb{Z}^n/M \oplus M$. For the APPROXFREQ problem and other related problems, it will suffice to consider only free automata. \[\Box\] Lemma 6 shows that the transition function $\oplus$ of a free automata can be represented as a linear mapping.

\[\text{There exist stream automata that use finite field arithmetic and consequently have torsion, for example} \quad \Box.\]
Lemma 6. Let $A_n$ be free automaton with kernel $M$. There exists a unique vector subspace $M^e$ of $\mathbb{R}^n$ of the smallest dimension containing $M$. The mapping $x + M \mapsto x + M^e$ is an injective mapping from $\mathbb{Z}^n/M$ to $\mathbb{R}^n/M^e$. If $\dim \mathbb{Z}^n/M = r$, then, there exists an orthonormal basis $V = [V_1, V_2]$ of $\mathbb{R}^n$ such that $\text{rank}(V_1) = r$, $\text{rank}(V_2) = n - r$, $M^e$ is the linear span of $V_2$ and $\mathbb{R}^n/M^e$ is the linear span of $V_1$. □

Proof. $\mathbb{Z}$ is a principal and entire ring. Since $\mathbb{Z}^n$ is a module over $\mathbb{Z}$, its sub-modules are free modules. Therefore, $M$ is a free module. Since $\mathbb{Z}^n/M$ is given to be free, $\mathbb{Z}^n$ is the direct sum of two free modules, $\mathbb{Z}^n = \mathbb{Z}^n/M \bigoplus M$. Therefore, both $M$ and $\mathbb{Z}^n/M$ have bases, say $B_1$ and $B_2$ whose union is a basis for $\mathbb{Z}^n$. Since, $\mathbb{Z}^n$ is a free module and has the standard $n$-dimensional basis $e_1, \ldots, e_n$, therefore, all bases of $\mathbb{Z}^n$ have the same dimension. Without loss of generality, therefore, let $B = [b_1, b_2, \ldots, b_n]$ be a basis of $\mathbb{Z}^n$ such that $B_2 = [b_1, \ldots, b_r]$ is a basis for $M$ and $B_1 = [b_{r+1}, \ldots, b_n]$ is a basis for $\mathbb{Z}^n/M$.

Let $M^e$ denote the span of $b_1, \ldots, b_r$ over $\mathbb{R}$. $M^e$ is obviously the smallest vector space over $\mathbb{R}$ that contains $M$, since, every vector space over $\mathbb{R}$ containing $M$ must contain the span of $b_1, \ldots, b_r$. Therefore, $\dim M^e \leq r$ and therefore, $\dim \mathbb{R}^n/M^e \leq n - r$ (same argument). However, the standard basis $\{e_1, \ldots, e_n\}$ is a basis of $\mathbb{Z}^n$ and therefore, $\dim M^e + \dim \mathbb{R}^n/M^e = n$. Hence, $\dim M^e = r$ and $\dim \mathbb{R}^n/M^e = n - r$. Further, $b_1, \ldots, b_n$ continues to be a basis for $\mathbb{R}^n$, of which $b_1, \ldots, b_r$ is a basis for $M^e$ and $b_{r+1}, \ldots, b_n$ is a basis for $\mathbb{R}^n/M^e$.

Consider the mapping $x + M \mapsto x + M^e$. Let $\bar{x}, \bar{y}$ denote the elements $x + M$ and $y + M$ of $\mathbb{Z}^n/M$. Suppose that $\bar{x} \neq \bar{y}$. Then, $x - y \notin M$. $x - y$ can be expressed uniquely as a linear combination of the basis elements.

$$x - y = \sum_{j=1}^n \alpha_j b_i, \quad \alpha_i \in \mathbb{Z}$$

Hence, $x - y$ has the same unique representation in the vector space over $\mathbb{R}^n$. Further, at least one of the coordinates $\alpha_1, \ldots, \alpha_r$ is non-zero, otherwise, $x - y$ would belong to $M$. Since, $x - y$ has the same representation in the vector space $\mathbb{R}^n$, $x - y$ is not in $M^e$. The mapping $x + M \mapsto x + M^e$ is therefore injective. Using standard Gram-Schmidt orthonormalization of $B_1$ and $B_2$ respectively viewed as defining vector sub-spaces over $\mathbb{R}$, we get $V_1$ and $V_2$. By the previous argument, $\text{rank}(V_1) = n - r$ and $\text{rank}(V_2) = r$. □

4 Frequency estimation

In this section, we present a space lower bound for APPROXFREQ($\epsilon$) using path-independent automaton. Recall that a stream automaton $A_n$ solves APPROXFREQ($\epsilon$), provided, after processing any input stream $\sigma$ with freq $\sigma = x$, $A_n$ returns a vector $\hat{x} \in \mathbb{R}^n$ satisfying $\text{err}(\hat{x}, x) = \frac{||\hat{x} - x||_\infty}{||x||_1} \leq \epsilon$. In general, if an estimation algorithm returns the same estimate $u$ for all elements of a set $S$, then, $\text{err}(u, S)$ is defined as $\max_{y \in S} \text{err}(u, y)$. Given a set $S$, let $\text{min}_{\ell_1}(S)$ denote the element in $S$ with the smallest $\ell_1$ norm: $\text{min}_{\ell_1}(S) = \text{argmin}_{y \in S} ||y||_1$.

Lemma 7. If $S \subset \mathbb{Z}^n$ and there exists $h \in \mathbb{R}^n$ such that $\text{err}(h, S) \leq \epsilon$, then $\text{err}(\text{min}_{\ell_1}(S), S) \leq 2\epsilon$.

Proof. Let $g$ denote $\text{min}_{\ell_1}(S)$ and $y \in S$. Since, $||g||_1 \leq ||y||_1$, by triangle inequality,

$$\text{err}(g, y) = \frac{||g - y||_\infty}{||y||_1} \leq \frac{||g - h||_\infty}{||y||_1} + \frac{||h - y||_\infty}{||y||_1} \leq \frac{||g - h||_\infty}{||g||_1} + \frac{||h - y||_\infty}{||y||_1} \leq \epsilon + \epsilon = 2\epsilon \quad \square$$
4.1 Frequency estimation using free automata

In this section, let $A_n$ be a free automaton with kernel $M$ that solves the problem $\text{APPROXFREQ}(\epsilon)$.

**Lemma 8.** Let $M$ be a sub-module of $\mathbb{Z}^n$. (1) if there exists $h$ such that $\text{err}(h, M) \leq \epsilon$, then, $\text{err}(0, M) \leq \epsilon$, and, (2) if $\text{err}(0, M) \leq \epsilon$ then $\text{err}(0, M_\epsilon) \leq \epsilon$.

**Proof (of Lemma 8 part (1)).** For any $y_i \in \mathbb{Z}$, $\max(|h_i - y_i|, |h_i + y_i|) \geq |y_i|$. Therefore,

\[
\max(\|y - h\|_\infty, \|h + y\|_\infty) \geq \|y\|_\infty.
\]

Let $y \in M$. Since, $M$ is a module, $-y \in M$. Thus,

\[
\text{err}(0, y) = \text{err}(0, -y) = \frac{\|y\|_\infty}{\|y\|_1} \leq \frac{1}{\|y\|_1} \max(\|y - h\|_\infty, \|h + y\|_\infty)
\]

\[
= \max(\text{err}(h, y), \text{err}(h, -y)) \leq \epsilon \quad \square
\]

**Proof (of Lemma 8 part (2)).** Let $z \in M_\epsilon$. Let $b_1, b_2, \ldots, b_r$ be a basis of the free module $M$. For $t > 0$, let $tz$ be expressed uniquely as $tz = \alpha_1 b_1 + \ldots + \alpha_r b_r$, where, $\alpha_i$'s belong to $\mathbb{R}$. Consider the vertices of the parallelopiped $P_{tz}$ whose sides are $b_1, b_2, \ldots, b_r$ and that encloses $tz$.

\[
P_{tz} = [\alpha_1]b_1 + [\alpha_2]b_2 + \ldots + [\alpha_n]b_n
\]

\[+ \{\beta_1 b_1 + \beta_2 b_2 + \ldots + \beta_r b_r \mid \beta_j \in \{0,1\}, j = 1, 2, \ldots, r\}\]

where, $[\alpha]$ denotes the largest integer smaller than or equal to $\alpha$. Since, $\ell_\infty$ is a convex function $\|tz\|_\infty \leq \|y\|_\infty$ for some $y \in P_{tz}$. Let $y = \sum_{j=1}^{r} \beta_j b_j$, for $\beta_j \in \{0,1\}, j = 1, 2, \ldots, r$. Then,

\[
\|y - tz\|_1 \leq \sum_{j=1}^{r} \|\beta_j - [\alpha_j]\| b_j \|
\]

\[
\text{or,} \quad \|tz\|_1 \geq \|y\|_1 - \sum_{j=1}^{r} \|b_j\|_1
\]

Therefore,

\[
\text{err}(0, tz) = \frac{\|tz\|_\infty}{\|tz\|_1} \leq \frac{\|y\|_\infty}{\|y\|_1 - \sum_{j=1}^{r} \|b_j\|_1}
\]

\[
\leq \left( \frac{\|y\|_1}{\|y\|_\infty} - \frac{\sum_{j=1}^{r} \|b_j\|_1}{\|y\|_\infty} \right)^{-1} \leq \left( \frac{1}{\epsilon} - \frac{\sum_{j=1}^{r} \|b_j\|_1}{\|y\|_\infty} \right)^{-1}
\]

where, the last step follows from the assumption that $y \in M$ and therefore, $\text{err}(0, y) = \frac{\|y\|_\infty}{\|y\|_1} \leq \epsilon$. The ratio $\sum_{j=1}^{r} \|b_j\|_1$ can be made arbitrarily small by choosing $t$ to be arbitrarily large. Thus, $\lim_{t\to\infty} \text{err}(0, tz) \leq \epsilon$. Since, $\text{err}(0, tz) = \frac{\|tz\|_\infty}{\|tz\|_1} = \frac{\|y\|_\infty}{\|y\|_1} = \text{err}(0, z)$, for all $t$, we have, $\text{err}(0, z) \leq \epsilon$. \quad \square

**Lemma 9.** Let $A_n$ be a free automaton that solves $\text{APPROXFREQ}(\epsilon)$ and has kernel $M$. Let $M^\epsilon$ be the smallest dimension subspace of $\mathbb{R}^n$ containing $M$. Let $V_1, V_2$ be a collection of vectors that forms an orthonormal basis for $\mathbb{R}^n$ such that $V_2$ spans $M^\epsilon$ and $V_1$ spans $\mathbb{R}^n / M^\epsilon$. Then, for $1/\sqrt{6n} < \epsilon \leq 1/8$, rank($V_1$) $\geq \sqrt{\frac{n}{2}}$. 
Proof. Since, $V_1$ has orthogonal columns
\[ \|V_1 V_1^T e_i\|_2^2 = \|V_i^T e_i\|_2^2 = (V_1 V_1^T)_{ii}. \]  
(5)

Therefore,
\[ \text{trace}(V_1 V_1^T) = \sum_{i=1}^n (V_1 V_1^T)_{ii} = \sum_{i=1}^n \|V_1 V_1^T e_i\|_2^2 \]

The trace of $V_1 V_1^T$ is the sum of the eigenvalues of $V_1 V_1^T$. Since, $V_1$ is orthogonal columns and has rank $\text{rank}(V_1)$, $V_1 V_1^T$ has eigenvalue 1 with multiplicity $\text{rank}(V_1)$ and eigenvalue 0 with multiplicity $n - \text{rank}(V_1)$. Thus, $\text{trace}(V_1 V_1^T) = \text{rank}(V_1) = r$ (say). It follows that
\[ r = \text{trace}(V_1 V_1^T) = \sum_{i=1}^n \|V_1 V_1^T e_i\|_2^2. \]  
(6)

Further,
\[ \sum_{i=1}^n \|V_1 V_1^T e_i\|_1 \leq \sum_{i=1}^n \|V_1 V_1^T e_i\|_2 \sqrt{n}, \quad \text{since, } \|x\|_1 \leq \|x\|_2 \sqrt{n} \]
\[ \leq \sqrt{n} \left( \sum_{i=1}^n \|V_1 V_1^T e_i\|_2^2 \right)^{1/2} n^{1/2}, \quad \text{by Cauchy-Schwartz inequality} \]
\[ = n \sqrt{k} \quad \text{by (6).} \]  
(7)

Let
\[ J = \{V_1 V_1^T e_i : 1 \leq i \leq n \text{ and } \|V_1 V_1^T e_i\|_2 \leq 3r/n\}, \]
\[ K = \{V_1 V_1^T e_i : 1 \leq i \leq n \text{ and } \|V_1 V_1^T e_i\|_1 \leq 3\sqrt{r}\}. \]

Therefore, by (6) and (7),
\[ |J| \geq \frac{2n}{3} \quad \text{and} \quad |K| \geq \frac{2n}{3}. \]

Hence, $J \cap K \neq \phi$, that is, there exists $i$ such that $\|V_1 V_1^T e_i\|_2 \leq (3r/n)^{1/2}$ and $\|V_1 V_1^T e_i\|_1 \leq 3\sqrt{r}$. Since, $e_j - V_1 V_1^T e_i = V_2 V_2^T e_i \in M$, therefore,
\[ \epsilon \geq \text{err}(e_j - V_1 V_1^T e_i, 0) = \frac{\|e_i - V_1 V_1^T e_i\|_\infty}{\|e_i - V_1 V_1^T e_i\|_1}. \]

Therefore,
\[ \|e_i - V_1 V_1^T e_i\|_\infty \leq \epsilon \|V_1 V_1^T e_i - e_i\|_1. \]  
(8)

By (5),
\[ (V_1 V_1^T e_i)_i = \|V_1 V_1^T e_i\|_2 \leq \frac{3r}{n} \]

Therefore,
\[ \|e_i - V_1 V_1^T e_i\|_\infty \geq \|(e_i - V_1 V_1^T e_i)_i\| = 1 - \|V_1 V_1^T e_i\|_2 \geq 1 - \frac{3r}{n}, \quad \text{by (5) and since } V_1 V_1^T e_i \in J. \]
Substituting in (\(N\)),

\[
1 - \frac{3r}{n} \leq \|e_i - V_1V_1^Te_i\|_\infty \leq \epsilon\|V_1V_1^Te_i - e_i\|_1 \leq \epsilon(\|V_1V_1^Te_i\| + 1),
\]

by triangle inequality \(\leq \epsilon(3\sqrt{n} + 1)\), since \(V_1V_1^Te_i \in K\).

Simplifying, \(r \geq \min \left( \frac{n}{6/1/(36\epsilon^2) - 1/9\epsilon} \right)\). Therefore, for \(1/\sqrt{6}n < \epsilon \leq \frac{1}{8}\), \(r \geq \frac{1}{72\epsilon}\). \(\square\)

**Lemma 10.** Let \(\frac{1}{6\sqrt{n}} \leq \epsilon < \frac{1}{8}\). Suppose \(A_n\) be a free automaton that uses \(s(n, m)\) bits on the work-tape to solve APPROXFREQ(\(\epsilon\)). Then, \(s(n, m) = \Omega\left( \frac{\log m}{\epsilon^2} \right)\).

**Proof.** Let \(M = \text{kernel of } A_n\). By Lemma \(9\), \(\text{rank}(V_1) = n - \dim M^e = \Omega\left( \frac{\epsilon}{\epsilon^2} \right)\). By Lemma \(10\), \(s(n, m) = \Omega((n - \dim M) \log m)\). Since \(\dim M = \dim M^e\), the result follows. \(\square\)

### 4.2 General path independent automata

We now show that for the problem APPROXFREQ(\(\epsilon\)), it is sufficient to consider free automata. Let \(A_n\) be a path-independent automaton that solves APPROXFREQ(\(\epsilon\)) and has kernel \(M\). Suppose that \(\mathbb{Z}^n/M\) is not free. Let \(M'\) be the module that removes the torsion from \(\mathbb{Z}^n/M\), that is,

\[
M' = \{ x \in \mathbb{Z}^n | \exists a \in \mathbb{Z}, a \neq 0 \text{ and } ax \in M \}. \tag{9}
\]

**Lemma 11.** \(\mathbb{Z}^n/M'\) is torsion-free.

**Proof (Of Lemma 11).** Suppose \(\tilde{y} = y + M'\) is a torsion element in \(\mathbb{Z}^n/M'\). Then, there exists \(b \in \mathbb{Z}\) and \(b \neq 0\) such that \(b\tilde{y} = by + M' \in M'\) or that \(by \in M'\). Therefore, there exists \(a \in \mathbb{Z}, a \neq 0\), such that \(by = ax\), for some \(x \in M\), or that, \(y = (b^{-1}a)x\) with \(b^{-1}a \neq 0\). Therefore, \(y \in M\). Hence, \(\mathbb{Z}^n/M'\) is torsion-free. \(\square\)

**Fact 12** Let \(b_1, b_2, \ldots, b_r\) be a basis of \(M'\). Then, \(\exists \alpha_1, \ldots, \alpha_r \in \mathbb{Z} - \{0\}\) such that \(\alpha_1b_1, \ldots, \alpha_r b_r\) is a basis for \(M\). Hence, \(M^e = (M')^e\).

**Proof (Of Fact 12).** It follows from standard algebra that the basis of \(M\) is of the form \(\alpha_1b_1, \ldots, \alpha_r b_r\). It remains to be shown that the \(\alpha_i\)'s are non-zero. Suppose that \(\alpha_1 = 0\). For any \(a \in \mathbb{Z}, a \neq 0\), suppose \(ax \in M\) and \(x \in M'\). Then, \(x\) has a unique representation as \(x = \sum_{j=1}^{r} x_j b_j\). Thus, \(ax = \sum_{j=1}^{r} (ax_j)b_j \in M\) and has the same representation in the basis \(\{\alpha_jb_j\}_{j=1, \ldots, n}\). Therefore, \(ax_1 = 0\) or \(x_1 = 0\) for all \(x \in M'\), which is a contradiction.

Let \(\{b_1, b_2, \ldots, b_r\}\) be a basis for \(M'\). Then, by the above paragraph, there exist non-zero elements \(\alpha_1, \ldots, \alpha_r\) such that \(\{\alpha_1b_1, \alpha_2b_2, \ldots, \alpha_r b_r\}\) is a basis for \(M\). Therefore, over reals, \((b_1, \ldots, b_r) = (\alpha_1b_1, \ldots, \alpha_r b_r)\). Thus, \(M^e = (M')^e\). \(\square\)

We show that if a path independent automaton with kernel \(M\) can solve APPROXFREQ(\(\epsilon\)), then a free automaton with kernel \(M' \supset M\) can solve APPROXFREQ(\(4\epsilon\)).

**Lemma 13.** Suppose \(A_n\) is a path independent automaton for solving APPROXFREQ(\(\epsilon\)) and has kernel \(M\). Then, there exists a free automaton \(B_n\) with kernel \(M'\) such that \(M' \supset M\), \(\mathbb{Z}^n/M'\) is free, and \(\text{err}(\min_{\epsilon_i}(x + M'), x) \leq 4\epsilon\).
Lemma 15. Let $M$ be the kernel of $A_n$ and let $M'$ be as defined in (9), so that $\mathbb{Z}^n/M'$ is free. For $x \in \mathbb{Z}^n$, define $h(x + M') = \min_{\ell_1}(x + M')$. Let $y \in x + M'$. Then, $y \in x_1 + M$ for some $x_1$. Let $\hat{y} = \text{output}_{A_n}(x_1 + M)$ denote the output of $A_n$ for an input stream with frequency in $x_1 + M$ (they all return the same value, since, $A_n$ is path independent and has kernel $M$) and let $y' = \min_{\ell_1}(x_1 + M)$. Let $h$ denote $h(x + M')$ and let $\hat{h} = \text{output}_{A_n}(h + M)$. Therefore,

$$
\text{err}(h, y) = \frac{\|y - h\|_\infty}{\|y\|_1} \leq \frac{\|y - \hat{y}\|_\infty}{\|y\|_1} + \frac{\|\hat{y} - y'\|_\infty}{\|y\|_1} + \frac{\|y' - h\|_\infty}{\|y\|_1}
$$

(10)

The first and the second terms above are bounded by $\epsilon$ as follows. The first term $\frac{\|y - \hat{y}\|_\infty}{\|y\|_1} = \text{err}(\hat{y}, y) \leq \epsilon$, since, $y \in x_1 + M$ and $\hat{y}$ is the estimate returned by $A_n$ for this coset. The second term

$$
\frac{\|\hat{y} - y'\|_\infty}{\|y\|_1} \leq \frac{\|\hat{y} - y'\|_\infty}{\|y'\|_1} = \text{err}(\hat{y}, y') \leq \epsilon
$$

since, $\|y'\|_1 \leq \|y\|_1$ and $y'$ lies in the coset $x_1 + M$. The third term in (10) can be rewritten as follows. By Lemma 11, $y' - h \in M'$ and $M' \subseteq M^e$. Therefore,

$$
\frac{\|y' - h\|_\infty}{\|y\|_1} \leq \frac{\|y' - h\|_\infty}{\|y'\|_1} \cdot \frac{\|y' - h\|_1}{\|y'\|_1}, \text{ since, } \|y'\|_1 \leq \|y\|_1
$$

$$
\leq \epsilon \cdot \frac{\|y'\|_1 + \|h\|_1}{\|y'\|_1} \text{ by Lemma } 8 \text{ and by triangle inequality}
$$

$$
\leq 2\epsilon, \text{ since, } \|h\|_1 \leq \|y'\|_1
$$

By (10), $\text{err}(h, y) \leq \epsilon + \epsilon + 2\epsilon = 4\epsilon$. The automaton $B_n$ with kernel $M'$ is constructed as in Lemma 5.

Lemma 14. Suppose $\frac{1}{24\sqrt{n}} \leq \epsilon < \frac{1}{3\sqrt{n}}$. Let $A_n$ be a path independent automaton that solves $\text{APPROXFREQ}(\epsilon)$. If $A_n$ has kernel $M$, then, $n - \dim M = \Omega\left(\frac{1}{\epsilon^2}\right)$.

Proof. By Lemma 13 there exists a free automaton $A'_n$ with kernel $M' \supset M$ that solves $\text{APPROXFREQ}(4\epsilon)$. Therefore, $n - \dim M \geq n - \dim M' = \Omega\left(\frac{1}{\epsilon^2}\right)$, by Lemma 10.

5 Path reversible automata

In this section, we show that given a path reversible automaton $A_n$, one can construct a path independent automaton $B_n$ that is an output restriction of $A_n$ and $\text{Space}(B_n, m) \leq \text{Space}(A_n, m) + O(\log n)$. Let $A_n$ be a path reversible automaton. For $f \in \mathbb{Z}^n$, define $\phi_{A_n}(f) = \{s \mid \exists \sigma \text{ s.t. } o \uplus \sigma = s \text{ and } \text{freq } \sigma = f\}$. The kernel of $A_n$ is defined as follows: $M = M_{A_n} = \{f \mid o \in \phi_{A_n}(f)\}$. Let $C = C(A_n)$ be the set of reachable configurations from the initial state $o$ of $A_n$ and let $C_m = C_m(A_n)$ denote the subset of $C(A_n)$ that are reachable from the initial state $o$ on input streams $\sigma$ with $|\sigma| \leq m$. Define a binary relation over $C$ as follows: $s \sim t$ if there exists $f \in \mathbb{Z}^n$ such that $s, t \in \phi_{A_n}(f)$.

Lemma 15. 1. $M$ is a sub-module of $\mathbb{Z}^n$.
2. If $f - g \in M$ then $\phi_{A_n}(f) = \phi_{A_n}(g)$, and, if $\phi_{A_n}(f) \cap \phi_{A_n}(g)$ is non-empty, then $f - g \in M$.
3. The relation $\sim$ over $C$ is an equivalence relation.
4. The map $[s] \mapsto f + M$, for $s \in \phi_{A_n}(f)$, is well-defined, 1-1 and onto.
Proof (Of Lemma 15, part 1.). Since the empty stream has frequency 0, 0 ∈ M. Suppose f ∈ M. There exists σ such that freq σ = f and o ⊕ σ = o. By path reversibility, o = o ⊕ σ ◦ σ∗ = o ⊕ σ∗. Since freq σ∗ = −freq σ = −f, therefore, −f ∈ M. Now suppose f, g ∈ M. Then there exists σ, τ such that freq σ = f, freq τ = g, o ⊕ σ = o and o ⊕ τ = o. Therefore, o ⊕ σ ◦ τ = o ◦ τ = o. Since, freq σ ◦ τ = freq σ + freq τ = f + g, therefore, f + g ∈ M. □

Proof (Of Lemma 15, part 2.). Suppose f = g + h, for some h ∈ M. Then, there exists σ such that o ⊕ σ = o and freq σ = h. Let a ∈ φA(n)(g) and let τ be a stream such that o ⊕ τ = a and freq τ = g. Then, o ⊕ σ ⊕ τ = o ⊕ τ = a, and freq σ ⊕ τ = freq σ + freq τ = h + g = f. Therefore, a ∈ φA(n)(f), or, φA(n)(g) ⊂ φA(n)(f). Reversing the roles of f and g, we have, φA(n)(f) ⊂ φA(n)(g), or that, φA(n)(f) = φA(n)(g). This proves the first assertion of the lemma. Conversely, Suppose a ∈ φA(n)(f) ∩ φA(n)(g). Then, there exist streams σ and τ such that freq σ = f, freq τ = g and o ⊕ σ = o ⊕ τ = a. By path reversibility, a ⊕ τr = o. Therefore, o ⊕ σ ◦ τr = a ◦ τr = o, and freq σ ◦ τr = freq σ + freq τr = f − g. Therefore, o ∈ φA(n)(f − g) and so f − g ∈ M. □

Proof (Of Lemma 15, part 3.). By definition, ~ is reflexive and symmetric. Suppose that s ∼ t and t ∼ u. Then, there exists f, g ∈ Zn such that s, t ∈ φA(n)(f) and t, u ∈ φA(n)(g). Therefore, t ∈ φA(n)(f) ∩ φA(n)(g). Hence, f − g ∈ M and so φA(n)(f) = φA(n)(g). Thus, s ∼ u. □

Proof (Of Lemma 15, part 4.). Suppose s ∈ φA(n)(f) ∩ φA(n)(g), then, f − g ∈ M, by Lemma 15, part 2, or that, f + M = g + M. Hence, the map is well-defined. Suppose [s] and [t] both map to f + M. Then, s, t ∈ φA(n)(f), and so s ∼ t and therefore, [s] = [t]. Hence the map is 1-1. For f ∈ Zn, φA(n)(f) is non-empty and for any s ∈ φA(n)(f), [s] maps to f + M, proving onto ness.

Let Bn be a path independent stream automaton whose configurations are the set of cosets of M and whose transition is defined as by the sum of the cosets, that is, f + (x + M) = (f + x) + M, constructed using Lemma 5. Its output on an input stream of A is defined as:

outputBn(σ) = choice {output of A in configuration s | s ∈ φA(n)(freq σ)}

where, choice S returns some element from its argument set S.

Lemma 16. Bn is an output restriction of A.

Proof. f + M = g + M if and only if φA(n)(f) = φA(n)(g). Therefore, outB(σ) is well-defined. Further, by definition of outB, outB(σ) = the output of A in some configuration s, where, s ∈ φA(n)(freq σ). Thus, Bn is an output restriction of An. □

We can now prove the main lemma of the section.

Lemma 17. Let A be a path reversible automaton with kernel M. Then, there exists a path independent automaton Bn with kernel M that is an output restriction of A such that log|Cm(A)| + O(log n) ≥ Space(Bn, m), for m ≥ 1.

Proof. Let Bn be constructed in the manner described above. By Lemma 16, Bn is an output restriction of A. Since the map [s] → f + M, for s ∈ φA(n)(f) is 1-1 and onto (Lemma 15, part 4), therefore, for every m, each reachable configuration of Bn after processing streams σ with freq σ ∈ [−m . . . m]n can be associated with a disjoint aggregate of configurations of Bn. The number of reachable configurations of Bn after processing streams with frequency in [−m . . . m]n is \(|x + M | x \in [−m . . . m]n\)|. Thus, |C(A)| ≥ |x + M | x \in [−m . . . m]n|].

By Lemma 5, Space(Bn, m) = log \(|x + M | x \in [−m . . . m]n| + O(log n). Combining, we obtain the statement of the lemma. □
Proof (Of Lemma 18.). Let
\[
\sigma = \mathcal{O}(\log n) \geq \text{Space}(B_n, m),
\]
for all \( m \geq 1 \). However, the arguments only use the property that the transition function \( \oplus_{A_n} \) is path reversible, and the fact that the subset of reachable configurations \( C_m(A_n) \) on streams of size at most \( m \) is finite. The argument is more general and also applies to computation performed by an infinite-state deterministic automaton in the classical sense that returns an output after it sees the end of its input, with set of states \( C \), initial state \( o \) and a path-reversible transition function \( \oplus'_{A_n} \). The above argument shows that such an automaton \( A_n \) can be simulated by a path-independent stream automaton \( B_n \) with finite control and additional space overhead of \( O(\log n) \) bits, such that \( B_n \) is an output-restriction of \( A_n \). We will use this observation in the next section.

6 Path non-reversible automata

In this section, we show that corresponding to every general stream automaton \( A_n \), there exists a path reversible automaton \( A'_n \) that is an output-restriction of \( A_n \), such that \( \text{Space}(A_n, m) \geq \log|C_m(A'_n)| \). By Lemma [17] corresponding to any path reversible automaton \( A'_n \), there exists an output-restricted and path independent automaton \( B_n \), such that \( \log|C_m(A'_n)| \geq \text{Space}(B_n, m) - O(\log n) \). Together, this proves a basic property of stream automata, namely, that, for every stream automaton \( A_n \), there exists a path-independent stream automaton \( B_n \) that is an output-restriction of \( A_n \) and \( \text{Space}(B_n, m) \leq \text{Space}(A_n, m) + O(\log n) \). We construct the path-reversible automaton \( A'_n \) only to the extent of designing a path-reversible transition function \( \oplus'_{A_n} \), a set of configurations \( C(A'_n) \) and specifying the output of \( A'_n \) if the end of the stream is met while at any \( s \in C(A'_n) \). As per the remarks at the end of the previous section, this is sufficient to enable the construction of the path-independent automaton \( B_n \) from \( A'_n \).

6.1 Defining reversible transition function from stream automata

In this section, we present detailed (existential) construction of constructing a reversible transition function \( \oplus' = \oplus'_{A_n} \) from a given general stream automaton \( A_n \) with transition function \( \oplus = \oplus_{A_n} \). Let \( C = C(A_n) \) denote the space of configurations of \( A_n \) and let \( C_m = C_m(A_n) \) denote the subset of \( C(A_n) \) that are reachable from \( o \) on input streams of size at most \( m \).

Consider a directed graph \( G = (C, E) \) where, \( C = C(A_n) \) is the set of vertices and there is a directed edge from \( s \) to \( t \) provided there is some stream \( \sigma \) such that \( \text{freq} \sigma = 0 \) and \( s \oplus \sigma = t \). Define the equivalence relation \( s \sim_G t \) if there is a directed path from \( s \) to \( t \) in \( G \) and vice-versa. Let \([s]_G\) denote the equivalence class to which a configuration \( s \) belongs. Define the equivalence class restricted to the vertices of \( C_m \) as \([s]_G \cap C_m \). An equivalence class \([s]_G \cap C_m \) that satisfies the property that for every stream \( \sigma \) with \( \text{freq} \sigma = 0 \) and \( s \oplus \sigma \in C_m \), we have \( s \oplus \sigma \in [s]_G \cap C_m \), are called terminal equivalence classes.

Lemma 18. For every \( m \geq 1 \) and \( u \in C_m \), there exists \( s = s(u) \) reachable from \( u \) in \( G_m \) such that \([s]_G \cap C_m \) is a terminal equivalence class.

Proof (Of Lemma [18]). Let \( u_0 \) be a vertex reachable from \( u \) in \( G_m \). If \([u_0]_G \cap C_m \) satisfies the property stated in the lemma, then, we are done. Otherwise, there exists \( \sigma \) such that \( \text{freq} \sigma = 0 \) and \( u_1 = u_0 \oplus \sigma \in C_m - [u_0]_G \cap C_m \). We now iteratively construct the sequence \([u_1]_G \cap C_m, [u_2]_G \cap C_m, \ldots \), in this manner. Suppose that two equivalence classes in this sequence
are the same, that is, suppose \([u_i] \sim_G \alpha_m = [u_j] \sim_G \alpha_m\). Then, there exists a directed path from \(u_i\) to \(u_j\) and vice-versa and therefore, \([u_i] \sim_G \alpha_m = \ldots = [u_j] \sim_G \alpha_m\), that is, the iteration terminates. Since, \(C_m\) is finite, the iterated sequence of equivalence classes of \(\sim_G \alpha_m\) terminates. The last equivalence class of this sequence satisfies the property of the lemma. \(\square\)

Define the mapping \(\alpha_m : C_m \to C_m\) as follows: \(\alpha_m(s)\) = some member of some terminal equivalence class reachable from \(s\) (for e.g., the member with least lexicographic value among all candidates). Fix \(s \in C\) and consider the sequence \(\{\alpha_m(s)\}_{m \geq 1}\). If this sequence is finite, then, one can define \(\alpha(s)\) to be a final element of the sequence. Otherwise, we use a standard technique of passing to the infinite case by associating \(s\) with ‘consistent’ infinite sequences \(\bar{s} = \{\alpha_m(s)\}_{m \geq 1}\).

**Lemma 19.** For \(s \in C\), \(\alpha(s) \oplus e_i \circ e_i = \alpha(s)\) and \(\alpha(s) \oplus e_i = \alpha(s)\).

**Proof (Of Lemma 19).** A configuration \(s\) is first identified with the infinite sequence, \(\bar{s} = \{\alpha_m(s)\}_{m \geq 1}\). Recall that the definition of \(\alpha_m(s)\) allows flexibility in the choice of a terminal class of \(\sim_G \alpha_m\). We now ensure that the choices are made in a consistent manner as follows. For each \(m\), there is a path \(P_m(s)\) from \(s\) to a vertex in the equivalence class \(\alpha_m(s)\). By consistent choices across \(m\), we mean that the \(P_{m+j}(s)\) is an extension of the path \(P_m(s)\), for each \(j > 0\), and for each \(s \in C\). From now

The transition function \(\oplus\) is defined in two steps. First, we define an intermediate function \(\oplus_1\).

\[ \bar{s} \oplus_1 e_i = \{\alpha_m(\alpha_m(s) \oplus e_i)\}_{m \geq 1} \]  

Sequences are allowed to have the undefined element \(\bot\), since, it is possible that \(s \not\sim C_m\) and hence \(\alpha_m(s)\) is not defined. However, if \(\alpha_m(s)\) is defined, then, \(\alpha_{m+j}(s)\) is defined, for all \(j > 0\). This implies that the undefined elements, if they occur, form a prefix of the sequence \(\bar{s}\).

We now attempt to prove Lemma 19 for the transition function \(\oplus_1\). Let \(m_0\) be the smallest \(m\) for which \(\alpha_m(s) \oplus e_i\) is well-defined. Then, for all \(m \geq m_0\), both \(\alpha_m(s) \oplus e_i\) and \(\alpha(\alpha_m(s) \oplus e_i) \oplus -e_i\) are also well-defined. The arguments in the finite case of Lemma 19 hold for each member \(m \geq m_0\). The same can be said for \(\alpha_m(s) \oplus -e_i\). Thus, the two sequences

\[ \{\alpha_m(s)\}_{m \geq 1} \text{ and } \{\alpha_m(\alpha_m(s) \oplus e_i) \oplus -e_i)\}_{m \geq 1} \]

differ at most in a finite prefix, where, the RHS sequence may have more \(\bot\) elements than the sequence on the LHS.

To resolve this problem, we define a relation \(\cong\) between pairs of infinite sequences.

\[ \{u_m\}_{m \geq 1} \cong \{v_m\}_{m \geq 1} \text{ if } u_m \text{ and } v_m \text{ differ in a finite initial prefix.} \]

A finite sequence \(u_1, \ldots, u_r\) is modeled as an infinite sequence \(u_1, \ldots, u_r, u_r, u_r, \ldots\) whose last term is repeated. It is straightforward to see that \(\cong\) is an equivalence relation on the family of sequences. It now follows that

\[ \{\alpha_m(s)\}_{m \geq 1} \cong \{\alpha_m(\alpha_m(s) \oplus e_i) \oplus -e_i)\}_{m \geq 1} . \]

For each configuration \(s\) in the original automaton, we associate it with \([s]\cong\) as follows.

\[ [s] \cong \{\alpha_m(s)\}_{m \geq 1} \]
The transition function $\oplus'$ is now defined as follows.

$$[s]_\leq \oplus e_i = \left[ \{ \alpha(n(s) \oplus e_i) \}_{m \geq 1} \right]_\leq$$

and

$$[s]_\leq \oplus -e_i = \left[ \{ \alpha(n(s) \oplus -e_i) \}_{m \geq 1} \right]_\leq$$

It now follows, by repeating the arguments in the previous paragraph, that

$$[s]_\leq \oplus' e_i \circ - e_i = [s]_\leq .$$

This proves Lemma 19 with $\alpha(s)$ defined as $\left[ \{ \alpha(n(s)) \}_{m \geq 1} \right]_\leq$. □

The map $s \mapsto \alpha(s)$ maps $s$ to a congruence class over the space of consistent infinite sequences. Define $C'_m = \{ \beta(s) \mid s \in C_m \}$. Therefore, $|C'_m| \leq |C_m|$ for all $m \geq 1$.

A path reversible automaton $A'_n$ is defined as follows. Initially $A'_n$ is in the state $\alpha(o)$. After reading a stream record (one of the $e_i$’s or $-e_i$’s), $A'_n$ uses the transition function $\oplus'$ instead of $\oplus$ to process its input. However, $s \oplus' \sigma = \alpha(s \oplus \sigma)$, where, $\alpha(t)$ is a set (possibly infinite) of states that cause $A_n$ to transition from configuration $t$ on some input $\sigma'$, with $\text{freq } \sigma' = 0$. Equivalently, this can be interpreted as if $\sigma'$ has been inserted into the input tape just after $A_n$ reaches the configuration $s$ and before it processes the next symbol—hence, $A'_n$ is an output-restriction of $A_n$ and is equally correct for frequency-dependent computations. This is the main idea of this construction. Thus, transitions of $\oplus'$ are equivalent to inserting some specifically chosen strings $\sigma_1, \sigma_2, \ldots$, each having $\text{freq } = 0$, after reading each letter (i.e., $\pm e_i$) of the input. The output of $A'_n$ on input stream $\sigma$ is identical to the output of $A_n$ on the stream $\sigma'$, where, $\sigma'$ is obtained by inserting zero frequency sub-streams into it. Therefore, $\text{freq } \sigma') = \text{freq } (\sigma)$ and $A'_n$ is an output restriction of $A_n$. By Lemma 19 the transition function $\oplus'$ is path reversible. Let $C' = C(A'_n)$ and $C'_m = C_m(A'_n)$. Since, $\alpha(s)$ is an equivalence class over $C(A_n)$, the map $s \mapsto \alpha(s)$ implies that $|C'_m| = |\{ \alpha(s) \mid s \in C_m \}| \leq |C_m|$. Starting from $A'_n$, one can construct a path independent automaton $B_n$ as per the discussion in Section 5. The arguments in this section do not show that the transition function $\oplus'$ can indeed by realized by a Turing machine that has only finite control. This is sufficient however, since, the path reversibility of $\oplus'$ is only used to allow the techniques of Section 5 to be applicable, and hence to be able to construct a coset-based path independent automaton. Since any coset based automaton can be realized using finite number of states in its finite control (Lemma 4), therefore, the final path-independent transition function is actually a stream automaton $B_n$.) Theorem 1 summarizes this discussion.

**Theorem 1 (Basic property of computations using stream automata).** For every stream automaton $A_n$, there exists a path-independent stream automaton $B_n$ that is an output-restriction of $A_n$ and $\text{Space}(B_n, m) \leq \text{Space}(A_n, m) + O(\log n)$.

**Proof.** Let $\oplus'$ be the transition function of the path-reversible automaton constructed as described above and let $B_n$ be the path-independent automaton obtained by translating $\oplus'$ using the procedure of Section 5. Let $C_m$ and $C'_m$ denote the number of reachable configurations of $A_n$ and $A'_n$, respectively, over streams with frequency vector in $[-m \ldots m]^n$. Let $s_A = s_A(n, m)$. Let $M$ be the kernel of $B_n$. Then,

$$|Q_A s_A 2^{s_A} \geq |C_m| \geq |C'_m| \geq |\{ x + M \mid x \in [-m \ldots m]^n \}| \geq (2m + 1)^{n-\dim M},$$

where, the last two inequalities follow from Lemma 17. Taking logarithms, $\text{Space}(A_n, m) \geq \log |\{ x + M \mid x \in [-m \ldots m]^n \}| \geq \text{Space}(B_n, m) - O(\log n)$, by Lemma 5. □
Theorem 2 (Lower bound for \textsc{ApproxFreq}(\epsilon)). Suppose that \(\frac{1}{24\sqrt{n}} \leq \epsilon < \frac{1}{32}\) and let \(A_n\) be a stream automaton that solves \textsc{ApproxFreq}(\epsilon). Then, \(\text{Space}(A_n, m) = \Omega\left(\frac{\log m}{\epsilon^2}\right) - O(\log n)\).

Proof. By Theorem 1 there exists a path independent automaton \(B_n\) that is an output-restriction of \(A_n\) and \(\text{Space}(A_n, m) \geq \text{Space}(B_n, m) - O(\log n)\). By Lemma 1, \(B_n\) solves \textsc{ApproxFreq}(\epsilon). If \(M\) is the kernel of \(B_n\), then by Lemma 4 \(\text{Space}(B_n) = \Omega((n - \dim M)(\log(2m + 1)))\). By Lemma 14 \(n - \dim M = \Omega(\epsilon^{-2})\). Thus,

\[
\text{Space}(A_n, m) = \Omega((n - \dim M) \log m) - O(\log n) = \Omega\left(\frac{\log m}{\epsilon^2}\right) - O(\log n) .
\]

Since, any path-independent automaton is arbitrarily mergeable (see text before Lemma 5), Theorem 1 implies that for any stream automaton \(A_n\), there exists an output-restricted automaton \(B_n\) such that \(\text{Space}(B_n, m) \leq \text{Space}(A_n, m) + O(\log n)\), and the state of \(B\) is arbitrarily mergeable, establishing the claim made in Section 1.

References

1. Noga Alon, Yossi Matias, and Mario Szegedy. “The space complexity of approximating frequency moments”. J. Comp. Sys. and Sc., 58(1):137–147, 1998.
2. P. Bose, E. Kranakis, P. Morin, and Y. Tang. “Bounds for Frequency Estimation of Packet Streams”. In Proc. SIROCCO, pages 33–42, 2003.
3. Moses Charikar, Kevin Chen, and Martin Farach-Colton. “Finding frequent items in data streams”. In Proc. ICALP, 2002, pages 693–703.
4. Graham Cormode and S. Muthukrishnan. “An Improved Data Stream Summary: The Count-Min Sketch and its Applications”. J. Algorithms, 55(1).
5. E. D. Demaine, A. López-Ortiz, and J. I Munro. “Frequency estimation of internet packet streams with limited space”. In Proc. ESA, pages 348–360, 2002.
6. P. Flajolet and G.N. Martin. “Probabilistic Counting Algorithms for Database Applications”. J. Comp. Sys. and Sc., 31(2):182–209, 1985.
7. S. Ganguly and Majumder A. “CR-precis: A Deterministic Summary Structure for Update Streams”. In Proc. Int’l Symp. on Algorithms, Probabilistic and Experimental Methodologies (ESCAPE), LNCS 4614, 2007.
8. S. Ganguly and A. Majumder. “Deterministic K-set Structure”. In Proc. ACM PODS, pages 280–289, 2006. Detailed version available from www.cse.iitk.ac.in/users/sganguly.
9. Sumit Ganguly. “Distributed deterministic approximation of vector sums”. Manuscript, November 2007.
10. Anna Gilbert, Y. Kotidis, S. Muthukrishnan, and Martin Strauss. “How to Summarize the Universe: Dynamic Maintenance of Quantiles”. In Proc. VLDB, pages 454–465, Hong Kong, August 2002.
11. R.M. Karp, S. Shenker, and C.H. Papadimitriou. “A Simple Algorithm for Finding Frequent Elements in Streams and Bags”. ACM TODS, 28(1):51–55, 2003.
12. J. Misra and Gries. D. “Finding repeated elements”. Sci. Comput. Programm., 2:143–152, 1982.