Evaluating Bayesian Networks via Data Streams

Andrew McGregor and Hoa T. Vu

University of Massachusetts, Amherst
{mcgregor,hvu}@cs.umass.edu

Abstract. Consider a stream of n-tuples that empirically define the joint distribution of n discrete random variables $X_1, \ldots, X_n$. Previous work of Indyk and McGregor [6] and Braverman et al. [1, 2] addresses the problem of determining whether these variables are n-wise independent by measuring the ℓ_p distance between the joint distribution and the product distribution of the marginals. An open problem in this line of work is to answer more general questions about the dependencies between the variables. One powerful way to express such dependencies is via Bayesian networks where nodes correspond to variables and directed edges encode dependencies. We consider the problem of testing such dependencies in the streaming setting. Our main results are:

1. A tight upper and lower bound of $\tilde{\Theta}(nk^d)$ on the space required to test whether the data is consistent with a given Bayesian network where k is the size of the range of each X_i and d is the max in-degree of the network.
2. A tight upper and lower bound of $\tilde{\Theta}(k^d)$ on the space required to compute any 2-approximation of the log-likelihood of the network.
3. Finally, we show space/accuracy trade-offs for the problem of independence testing using ℓ_1 and ℓ_2 distances.

1 Introduction

The problem of testing n-wise independence in data streams has attracted recent attention in streaming algorithms literature [1, 2, 6]. In that problem, the stream consists of a length m sequence of n-tuples that empirically defines a joint distribution of n random variables $X_1, X_2, \ldots, X_n$ where each X_i has range $[k] := \{1, 2, \ldots k\}$. Specifically, the stream defines the joint probability mass function (pmf):

$$\mathcal{P}(x_1, \ldots, x_n) = \mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) := \frac{f(x_1, x_2, \ldots, x_n)}{m} , \quad (1)$$

where $f(x_1, x_2, \ldots, x_n)$ is the number of tuples equal to $(x_1, x_2, \ldots, x_n)$. The marginal probability of a subset of variables $\{X_j\}_{j \in S}$ is defined as:

$$\mathbb{P}(X_j = x_j \ \ \forall j \in S) := \sum_{x_\ell \in [k] \text{ for all } \ell \notin S} \mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) .$$

The goal of the previous work was to determine whether this distribution is close to being a product distribution or equivalently, whether the corresponding random variables

are close to being independent by estimating:

$$\Big(\sum_{x_1,\ldots,x_n \in [k]} \big|\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) - \mathbb{P}(X_1 = x_1) \ldots \mathbb{P}(X_n = x_n)\big|^p \Big)^{1/p}$$
$$:= \Big\| \mathbb{P}(X_1, \ldots, X_n) - \mathbb{P}(X_1) \ldots \mathbb{P}(X_n) \Big\|_p := \mathcal{E}_p(\emptyset).$$

However, it is natural to ask more general questions about the dependencies between the variables, e.g., can we identify an X_i such that the other random variables are independent conditioned on X_i or whether there is an ordering $X_{\sigma(1)}, X_{\sigma(2)}, X_{\sigma(3)}, \ldots$ such that $X_{\sigma(i)}$ is independent of $X_{\sigma(1)}, X_{\sigma(2)}, \ldots, X_{\sigma(i-2)}$ conditioned on $X_{\sigma(i-1)}$.

The standard way to represent such dependencies is via Bayesian networks. A Bayesian network is an acyclic graph G with a node X_i corresponding to each variable X_i along with a set of directed edges E that encode a factorization of the joint distribution. Specifically, if $\mathrm{Pa}(X_i) = \{X_j : (X_j \to X_i) \in E\}$ are the parents of X_i in G then the Bayesian network represents the assertion that for all $x_1, x_2, \ldots, x_n$, the joint distribution can be factorized as follows:

$$\mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \prod_{i=1}^{n} \mathbb{P}(X_i = x_i | X_j = x_j \; \forall \; X_j \in \mathrm{Pa}(X_i)) \; .$$

For example, $E = \emptyset$ corresponds to the assertion that the X_i are fully independent whereas the graph on nodes $\{X_1, X_2, X_3\}$ with directed edges $X_1 \to X_2, X_1 \to X_3$ corresponds to the assertion that X_2 and X_3 are independent conditioned on X_1.

Bayesian networks have been extensively studied and applied in artificial intelligence, machine learning, data mining, and other areas. In these applications the focus is typically on Bayesian networks where d is small, since we wish to be able to compactly represent the joint distribution through local conditional probability distributions.

In this paper we consider the problem of evaluating how well the observed data fits a Bayesian network. The data stream of tuples in $[k]^n$ and a Bayesian network G defines an empirical distribution $\mathcal{P}_G$ with the pmf:

$$\mathcal{P}_G(x_1, \ldots, x_n) := \prod_{i=1}^{n} \mathbb{P}(X_i = x_i | X_j = x_j \; \forall \; X_j \in \mathrm{Pa}(X_i)), \tag{2}$$

where

$$\mathbb{P}(X_i = x_i | X_j = x_j \text{ for all } j \in \mathrm{Pa}(X_i)) = \frac{\mathbb{P}(X_i = x_i)}{\mathbb{P}(X_j = x_j, \; \forall X_j \in \mathrm{Pa}(X_i))}. \tag{3}$$

is just the fraction of tuples whose ith coordinate is x_i amongst the set of tuples whose jth coordinate is x_j for all $X_j \in \mathrm{Pa}(X_i)$. We then define the error of G to be the ℓ_p norm, for $p \in \{1, 2\}$, of the difference between the joint distribution and the factorization $\mathcal{P}_G$:

$$\mathcal{E}_p(G) := \big(\sum_{x_1,\ldots,x_n \in [k]} |\mathcal{P}(x_1, \ldots, x_n) - \mathcal{P}_G(x_1, \ldots, x_n)|^p \big)^{\frac{1}{p}} := \|\mathcal{P} - \mathcal{P}_G\|_p.$$

Clearly, if the factorization implied by G is valid then $\mathcal{E}_p(G) = 0$. More generally, if $\mathcal{E}_p(G)$ is small then we consider the factorization to be close to valid. The use of ℓ_p distance to measure "closeness" was considered previously in the Bayesian network literature [7, 11]. However, the space required to compute these measures was considered a major drawback because it was assumed that it would be necessary to explicitly store the full joint distribution whose space complexity is $O(k^n)$. Our results show that this is not the case. Note that when G is the empty graph, $\mathcal{E}_p(\emptyset)$ is the quantity measured in [1, 2, 6].

In many applications, data comes in a streaming fashion. When it comes to very large data volume, it is important to maintain a data structure that uses small memory and estimates different statistics about the data accurately at the same time. As the space requirement to measure the accuracy of Bayesian networks is as large as $O(k^n)$ and as the size of our data set m increases, our problem of evaluating Bayesian networks via data streams with small memory is of considerable importance.

1.1 Our Results

Here, and henceforth we use k, n, d and m to denote the range of the variables, the number of the variables, the maximum in-degree of the network and the length of the stream respectively.

1. *Testing and Estimating ℓ_p Accuracy.* For any Bayesian network G, we present a single-pass algorithm using $\tilde{\Theta}(nk^d)$ space[1] for the problem of testing whether the data is consistent with G, i.e., $\mathcal{E}_p(G) = 0$. We prove a matching lower bound showing that the dependence on n, k, and d is optimal. We also present a $\tilde{O}(\varepsilon^{-2}nk^{d+1})$-space algorithm for estimating $\mathcal{E}_p(G)$ up to a $(1 + \varepsilon)$ factor. The lower bound is based on the Local Markov Property, a result from Bayesian Networks literature, and a reduction from communication complexity.
2. *Estimating Log-Likelihood.* Next, we present a single-pass $\tilde{O}(nk^d)$-space algorithm that estimates the log-likelihood of a given network. We also prove a lower bound of $\Omega(k^d)$ for any factor 2 approximation of this quantity. As an application, we can find the branching tree network that approximately maximizes the log-likelihood of the observed streaming data in space $\tilde{O}(n^2k)$ with $O(n^2)$ post-processing time. Our algorithm is based on the Chow-Liu tree [4] construction.
3. *Trade-offs for Independence Testing.* We revisit the problem of independence testing in Section 5 and present space/accuracy trade-offs for estimating $\mathcal{E}_p(\emptyset)$. Specifically, for $p = 1$, we can achieve an $(n-1)/t$-approximation for any constant $1 \leq t < n/2$ using $O(\operatorname{poly} n)$ space compared to the $(1 \pm \varepsilon)$-approximation algorithm in [2] with space that is doubly-exponential in n. For $p = 2$, we present an $O(\operatorname{poly} n)$-space algorithm with additive error compared to the $O(3^n)$-space algorithm in [1] with multiplicative error.

1.2 Notation

$A \perp B \mid C$ denotes the assertion that random variables A, B are independent conditioned on C, i.e., $\mathbb{P}(A = a, B = b|C = c) = \mathbb{P}(A = a|C = c)\mathbb{P}(B = b|C = c)$ for all

[1] $\tilde{O}$ omits all poly-logarithmic factors of m, n, and k.

a, b, c in the range of A, B, C. $\mathrm{Pa}(X_i)$ denotes the set of variables that are parents of X_i and $\mathrm{ND}(X_i)$ denotes the set of variables that are non-descendants of X_i, other than $\mathrm{Pa}(X_i)$. If $X_1, \ldots, X_n \in [k]$ then we use $(X_1, \ldots, X_n)$ denote a tuple of n variables in $[k]^n$ or equivalently a single variable in the range $[k^n]$.

2 Algorithms for Estimating $\mathcal{E}_p(G)$

In this section, we present approximation algorithms for estimating $\mathcal{E}_p(G)$ for an arbitrary Bayesian network G and a more efficient algorithm just to test if $\mathcal{E}_p(G) = 0$.

2.1 $(1+\varepsilon)$-Approximation using $\tilde{O}(nk^{d+1})$ Space

We first note that the factorized distribution $\mathcal{P}_G$ can be computed and stored exactly in $O(nk^{d+1} \log m)$ bits since, by Eq. (1) and Eq. (3), it suffices to compute

$$\frac{\sum_{\mathbf{a} \in [k]^n \,:\, a_j = x_j \;\forall j \;s.t\; X_j \in \{X_i\} \cup \mathrm{Pa}(X_i)} f(\mathbf{a})}{\sum_{\mathbf{a} \in [k]^n \,:\, a_j = x_j \;\forall j \;s.t\; X_j \in \mathrm{Pa}(X_i)} f(\mathbf{a})} \,.$$

for each $i \in [n]$ and each of at most k^{d+1} combinations of values for X_i and $\mathrm{Pa}(X_i)$. Given this observation, it is straightforward to approximate $\mathcal{E}_p(G)$ given any data stream "sketch" algorithm that returns a $(1+\varepsilon)$ estimate for the ℓ_p norm of a vector v. Kane et al. [10] presented such as algorithm that uses space that is logarithmic in the dimension of the vector.

Specifically, we apply the algorithm on a vector v defined as follows. Consider v to be indexed as $[k] \times [k] \times \ldots \times [k]$. On the arrival of tuple $(x_1, \ldots, x_n)$, we increment the coordinate corresponding to $(x_1, \ldots, x_n)$ by $1/m$. At the end of the stream, v encodes the empirical joint distribution. For each $(x_1, \ldots, x_n)$, we now decrement the corresponding coordinate by $\mathcal{P}_G(x_1, \ldots, x_n)$. At this point, $v_{x_1,\ldots,x_n} = \mathcal{P}(x_1, \ldots, x_n) - \mathcal{P}_G(x_1, \ldots, x_n)$ and hence the ℓ_p norm of v is $\mathcal{E}_p(G)$. Hence, returning the estimate from the algorithm yields a $1+\varepsilon$ approximation to $\mathcal{E}_p(G)$ as required.

Note that this simple approach also improves over existing work [2] on the case of measuring $\ell_p(G)$ when G has no edges (i.e., measuring how far the data is from independent) unless n is very small compared to k. The space used in previous work is doubly-exponential in n but logarithmic in k whereas our approach uses $\tilde{O}(nk)$ space and hence, our approach is more space-efficient unless $k > 2^{n^n}/n$.

Theorem 1. *There exists a single-pass algorithm that computes $(1 \pm \varepsilon)\,\mathcal{E}_p(G)$ with probability at least $1-\delta$ using $\tilde{O}(\varepsilon^{-2} k^{d+1} n \log \delta^{-1})$ space.*

2.2 $2n$-Approximation using $\tilde{O}(\mathrm{poly}(n)k^d)$ Space

We now give an alternative algorithm with a weaker approximation guarantee but requires a smaller space in terms of k and d.

Theorem 2. *There exists a single-pass $\tilde{O}(\mathrm{poly}(n) \cdot k^d)$-space algorithm that computes an $O(n)$-approximation of $\mathcal{E}_1(G)$ with probability at least $1-\delta$.*

We first briefly describe the algorithm. Without loss of generality, assume X_n, X_{n-1}, $\ldots$, X_1 form a topological order in G. Such an order must always exist since G is acyclic. Let $\mathbf{X}(i,n)$ denote $(X_i,\ldots,X_n)$.

1. For each $i \in [n-1]$, compute a $(1+\varepsilon)$-factor approximation of
$$v_i := \Big|\mathbb{P}(\mathbf{X}(i,n)) - \mathbb{P}(X_i|\operatorname{Pa}(X_i))\mathbb{P}(\mathbf{X}(i+1,n))\Big|.$$
We shall explain how to get the approximation shortly.
2. Return the sum of the estimators above.

Proof. We start by showing that $\mathcal{E}_1(G) \le \sum_{i=1}^{n-1} v_i \le 2n\,\mathcal{E}_1(G)$. The first inequality is derived as follows.

$$\begin{aligned}
\mathcal{E}_1(G) &\le \Big|\mathbb{P}(\mathbf{X}) - \mathbb{P}(X_1|\operatorname{Pa}(X_1))\mathbb{P}(\mathbf{X}(2,n))\Big| \\
&\quad + \Big|\mathbb{P}(X_1|\operatorname{Pa}(X_1))\mathbb{P}(\mathbf{X}(2,n)) - \mathbb{P}(X_1|\operatorname{Pa}(X_1))\mathbb{P}(X_2|\operatorname{Pa}(X_2))\mathbb{P}(\mathbf{X}(3,n))\Big| \\
&\quad + \ldots + \Big|\prod_{i=1}^{n-2}\big(\mathbb{P}(X_i|\operatorname{Pa}(X_i))\mathbb{P}(X_{n-1},X_n) - \prod_{i=1}^{n}\mathbb{P}(X_i|\operatorname{Pa}(X_i))\Big| \qquad (4)\\
&= \sum_{i=1}^{n-1}\Big|\mathbb{P}(\mathbf{X}(i,n)) - \mathbb{P}(X_i|\operatorname{Pa}(X_i))\mathbb{P}(\mathbf{X}(i+1,n))\Big| = \sum_{i=1}^{n-1} v_i.
\end{aligned}$$

The first inequality follows from the triangle-inequality. For each ith term in Equation (4), we can factor out $\{\mathbb{P}(X_j|\operatorname{Pa}(X_j))\}_{j\in[i-1]}$ which sums (over X_j) to 1 as the inner factors do not involve X_j.

Next, we show that $v_i \le 2\,\mathcal{E}_1(G)$. By the triangle-equality we have:

$$\begin{aligned}
v_i &\le \Big|\mathbb{P}(\mathbf{X}(i,n)) - \prod_{j\ge i}\mathbb{P}(X_j|\operatorname{Pa}(X_j))\Big| \\
&\quad + \Big|\prod_{j\ge i}\mathbb{P}(X_j|\operatorname{Pa}(X_j)) - \mathbb{P}(X_i|\operatorname{Pa}(X_i))\mathbb{P}(\mathbf{X}(i+1,n)\Big|.
\end{aligned}$$

To bound each term on the right, we first introduce the following notation:

$$g_k(\mathbf{x}) = \mathbb{P}(X_k = \mathbf{x}_k | X_j = \mathbf{x}_j \text{ for all } X_j \in \operatorname{Pa}(X_k))$$

Then,

$$\begin{aligned}
&\Big|\mathbb{P}(\mathbf{X}(i,n)) - \prod_{j\ge i}\mathbb{P}(X_j|\operatorname{Pa}(X_j))\Big| \\
&= \sum_{\mathbf{b}\in[k]^{n-i+1}}\Big|\sum_{\mathbf{a}\in[k]^{i-1}}\mathbb{P}(\mathbf{X}(1,i-1)=\mathbf{a},\mathbf{X}(i,n)=\mathbf{b}) - \Big(\sum_{\mathbf{a}\in[k]^{i-1}}\prod_{1\le q<i} g_q(\mathbf{ab})\Big)\cdot\prod_{j\ge i} g_j(\mathbf{ab})\Big)\Big| \\
&\le \sum_{\substack{\mathbf{a}\in[k]^{i-1}\\ \mathbf{b}\in[k]^{n-i+1}}}\Big|\mathbb{P}(\mathbf{X}=\mathbf{ab}) - \prod_{1\le q<i} g_q(\mathbf{ab})\cdot\prod_{j\ge i} g_j(\mathbf{ab})\Big| = \mathcal{E}_1(G).
\end{aligned}$$

The second term is equal to $|\prod_{j\geq i+1}\mathbb{P}(X_j|\,\mathrm{Pa}(X_j))-\mathbb{P}(\mathbf{X}(i+1,n)|$ which is of similar form as the first term and can be upper bounded by $\mathcal{E}_1(G)$ similarly. We approximate each v_i as follows. For each $c\in[k^d]$, we have:

$$v_i(c)=\mathbb{P}(\mathrm{Pa}(X_i)=c)\Big|\mathbb{P}(\mathbf{X}(i,n)|\mathrm{Pa}(X_i)=c) \\ -\mathbb{P}(X_i|\,\mathrm{Pa}(X_i)=c)\mathbb{P}(\mathbf{X}(i+1,n)|\,\mathrm{Pa}(X_i)=c)\Big|.$$

We can compute $\mathbb{P}(\mathrm{Pa}(X_i)=c)$ exactly and approximate $v_i(c)$ using Theorem 9. Because $v_i=\sum_{c\in[k^d]}v_i(c)$, to get an estimate for v_i, we simply take the sum of the estimates for each $v_i(c)$. Since we need to do this for all $i\in[n],c\in[k^d]$, the space usage is $\tilde{O}(\mathrm{poly}(n)\cdot k^d)$.

2.3 Decision problem

We now show that testing $\mathcal{E}_p(G)=0$ can indeed be done in space that is tight with the lower bound in terms of n,k,d.

Definition 1. *A Bayesian network G with vertices $X_1,..,X_n$ satisfies the* Local Markov Property *if $X_i\perp\mathrm{ND}(X_i)\mid\mathrm{Pa}(X_i)$ for all $i\in[n]$.*

We rely on the following theorem. Its proof can be found in many Bayesian networks literature such as [8].

Theorem 3. *(Local Markov Property) Any given Bayesian network G satisfies $\mathcal{E}_p(G)=0$ iff it satisfies the Local Markov Property.*

The idea is to check the Local Markov Property for each variable in the network. However, to match the lower bound, we also need to resolve a subtle issue regarding storing the random vectors.

Theorem 4. *There exists an $\tilde{O}(k^dn)$-space single-pass algorithm that tests $\mathcal{E}_p(G)=0$ with probability at least $1-\delta$.*

Proof. For each X_i, because $|\,\mathrm{ND}(X_i)|\leq n$, $\mathrm{ND}(X_i)$ can be viewed as a single variable that takes at most k^n different values. We need to check if:

$$\eta(i):=\Big\|\mathbb{P}(X_i,\mathrm{ND}(X_i)|\,\mathrm{Pa}(X_i))-\mathbb{P}(X_i|\,\mathrm{Pa}(X_i))\mathbb{P}(\mathrm{ND}(X_i)|\,\mathrm{Pa}(X_i))\Big\|_2=0.$$

Call this testing algorithm $\mathcal{A}_i$. We define $\eta(i,c)$ to be the distance above with $\mathrm{Pa}(X_i)=c$. We have $\eta(i)=\sum_{c\in[k]^{|\,\mathrm{Pa}(X_i)|}}\eta(i,c)$. For any fixed c, we can test $\eta(i,c)=0$ by running the algorithm from Theorem 9.

For some $S\subseteq\{X_1,\ldots,X_n\}$ and $\mathbf{x}\in[k]^n$, let x_S denote the tuple of $\{x_j:X_j\in S\}$. The algorithm in Theorem 9 incrementally maintains the following sketches:

$$t_1=\sum_{a\in[k],b\in[k]^{|\,\mathrm{ND}(X_i)|},\mathbf{x}:x_i=a,x_{\mathrm{ND}(X_i)}=b,x_{\mathrm{Pa}(X_i)}=c}f(\mathbf{x})\gamma_a\lambda_b$$

$$t_2=\sum_{a\in[k],\mathbf{x}:x_i=a,x_{\mathrm{Pa}(X_i)}=c}f(\mathbf{x})\gamma_a\ \text{ and }\ t_3=\sum_{b\in[k]^{|\,\mathrm{ND}(X_i)|},\mathbf{x}:x_{\mathrm{ND}(X_i)}=b,x_{\mathrm{Pa}(X_i)}=c}f(\mathbf{x})\lambda_b$$

where $\lambda, \gamma \in \{-1, 1\}^{k^n}$ are 4-wise independent vectors. The space required to store these vectors is $O(\log k^n) = O(n \log k)$. It can be shown that [1, 6]:

$$\mathbb{E}[(\frac{t_1}{m} - \frac{t_2 t_3}{m^2})^2] = \eta(i,c)^2 \text{ and } \mathbb{V}ar[(\frac{t_1}{m} - \frac{t_2 t_3}{m^2})^2] \leq 9\eta(i,c)^4.$$

Hence, to have a factor 10 approximation of the distance that tests if $\eta(i, c) = 0$ with probability at least $1-\delta/(k^d n)$, we need to use $O(\log k^d + \log \delta^{-1} + \log n)$ independent λ, γ's in parallel and take the median of the estimators. We need to do this for all $c \in [k]^{|\operatorname{Pa}(X_i)|}$. Run $\mathcal{A}_i$ for all $i \in [n]$. *The key observation* is that all $\mathcal{A}_i$'s may use the same set of these 4-wise independent vectors. So the total space to run $\mathcal{A}_1, \ldots, \mathcal{A}_n$ is:

$$O(\underbrace{nk^d(\log k^d + \log \delta^{-1} + \log n) \log m}_{\text{space to store the sketches}} + \underbrace{k^d(\log k^d + \log \delta^{-1} + \log n) n \log k)}_{\text{space to store the random vectors}} = \tilde{O}(nk^d)\ .$$

By the union bound, we can tell if there is an X_i that does not satisfy the local Markov property with probability at least $1 - \delta$ in the space that is optimal up to a polylogarithmic factor.

3 Lower bounds for estimating $\mathcal{E}_p(G)$

Next, we show that the decision algorithm and the approximation algorithm above are optimal and near-optimal respectively. It has been shown that independence testing via ℓ_p distance can be done in $O(\operatorname{polylog} k)$ space. The open question we are trying to answer is whether it is still possible to test more general dependencies in $O(\operatorname{polylog} k)$ space. Unfortunately, the answer is, in general, no. We first prove that for testing whether two variables are perfectly independent given the third variable, any constant-pass streaming algorithm requires $\Omega(k)$ space.

The proofs of our lower bounds use the standard technique of reducing from a communication complexity problem. In particular, we consider the disjointness problem where Alice and Bob each have a string $x \in \{1, 2\}^k$ and $y \in \{1, 2\}^k$ respectively and want evaluate $\operatorname{DISJ}(x, y)$ where

$$\operatorname{DISJ}(x,y) = \begin{cases} 0 & \text{if there exists } i \text{ such that } x_i = y_i = 1 \\ 1 & \text{otherwise} \end{cases}$$

A classic result [9] shows that any (randomized) protocol with constant number of rounds for this problem requires $\Omega(k)$ bits to be communicated. The following remark is useful in our reduction.

Lemma 1. *Given a stream of two binary samples in the format (A, B) as $(a, 2), (2, b)$. Then, A, B are independent iff a, b are not both equal 1.*

Proof. If $a = b = 1$, then $\mathbb{P}(A = 1, B = 2) = 0.5 \neq \mathbb{P}(A = 1)\mathbb{P}(B = 2) = 0.5 \times 0.5 = 0.25$. Otherwise, one can easily check that $\mathbb{P}(A, B) = \mathbb{P}(A)\mathbb{P}(B)$.

Proposition 1. *There exists a network G such that any constant-pass algorithm that decides if $\mathcal{E}_p(G) = 0$ with probability at least 2/3 requires $\Omega(k^d)$ space.*

Proof. Consider the Bayesian network G with vertices $X_1, \ldots, X_d, Y, Z$ where each X_i is a parent of $X_1, X_2, \ldots, X_{i-1}, Y$, and Z. Let $\mathbf{X} = (X_1, \ldots, X_d)$. Then,

$$\begin{aligned}\mathcal{E}_p(G) &= \left\| \mathbb{P}(Y, Z|\mathbf{X})\mathbb{P}(\mathbf{X}) - \mathbb{P}(Y|\mathbf{X})\mathbb{P}(Z|\mathbf{X})\big(\prod_{i=1}^{d} \mathbb{P}(X_i|X_{i+1}, \ldots, X_d)\big) \right\|_p \\ &= \left\| \mathbb{P}(Y, Z|\mathbf{X})\mathbb{P}(\mathbf{X}) - \mathbb{P}(Y|\mathbf{X})\mathbb{P}(Z|\mathbf{X})\mathbb{P}(\mathbf{X}) \right\|_p . \end{aligned} \tag{5}$$

We make the reduction from DISJ where Alice and Bob, with bit strings a and b of length k^d, generate the stream S_A and S_B of $(Y, Z, \mathbf{X})$-tuples respectively:

$$S_A = \{(a_1, 2, \mathbf{x}) : \mathbf{x} \in [k]^d\} \quad , \quad S_B = \{(2, b_1, \mathbf{x}) : \mathbf{x} \in [k]^d\} .$$

By Equation (5), we have that $\mathcal{E}_p(G) = 0$ iff $Y \perp Z|\{\mathbf{X} = c\}$ for all $c \in [k]^d$. By Lemma 1, this is satisfied iff $\text{DISJ}(a, b) = 1$. Therefore, any constant-pass algorithm that decides if $\mathcal{E}_p(G) = 0$ requires $\Omega(k^d)$ space.

We now construct a more sophisticated reduction to incorporate n in the lower bound.

Theorem 5. *There exists a Bayesian network G such that any constant-pass algorithm that determines if $\mathcal{E}_p(G) = 0$ with probability at least 2/3 requires $\Omega(nk^d)$ space.*

Proof. Without loss of generality, assume n is a power of 2. Let $x \in \{1, 2\}^{nk}, y \in \{1, 2\}^{nk}$ be an instance of DISJ where it be convenient to index x and y by $[n] \times [k]$. The Bayesian network we consider is balanced binary tree with leaves A_1, B_1, A_2, $B_2, \ldots, A_n$, B_n and internal nodes R_i^j where R_i^1 in the parent of A_i and B_i and R_i^j is the parent of R_{2i-1}^{j-1} and R_{2i}^{j-1} for $j > 1$. The root node is $R_1^{\log n+1}$. See Figure 1. The variables R_i^j will take $3k$ different values and it will be convenient to index these values as $[3] \times [k]$. The leaf variables take either the value 1 or 2.

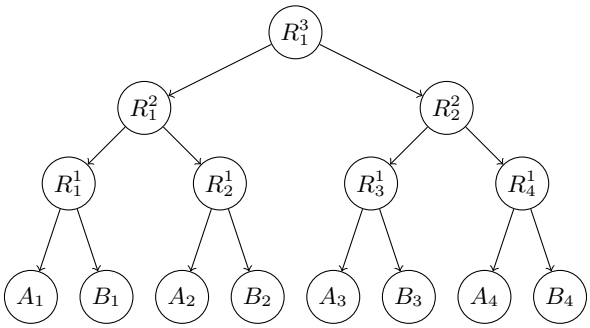

Fig. 1. Construction for $n = 4$

Alice generates a stream that defines samples from the joint distribution based on x. Each sample generated satisfies the following criteria and all distinct samples that obey this criteria are generated:

1. $R_1^{\log n+1} \in \{(1,z),(2,z) : z \in [k]\}$.
2. If $R_i^j = (1,z)$ for $j > 1$:
   - The left child $R_{2i-1}^{j-1} \in \{(1,z),(2,z)\}$ and the right child $R_{2i}^{j-1} = (3,z)$.
3. If $R_i^j = (2,z)$ for $j > 1$:
   - The left child $R_{2i-1}^{j-1} = (3,z)$ and the right child $R_{2i}^{j-1} \in \{(1,z),(2,z)\}$.
4. If $R_i^j = (3,z)$ for $j > 1$:
   - Both the values for the children R_{2i-1}^{j-1} and R_{2i}^{j-1} are $(3,z)$.
5. If $R_i^1 \in \{(1,z),(2,z)\}$:
   - The values for the children are $A_i = x_{i,z}$, $B_i = 2$
6. If $R_i^1 = (3,z)$:
   - The values for the children are $A_i = 2$, $B_i = 2$

Bob then generates a series samples in a similar manner except that Rule 5 becomes: If $R_i^1 \in \{(1,z),(2,z)\}$, then $A_i = 2$, $B_i = y_{i,z}$.

Note that each sample defined by either Alice or Bob specifies a path from the root to a pair A_i, B_i as following: Starting from the root, if the current node's value is equal to $(1,z)$, then go to its left child; on the other hand, if its value is equal to $(2,z)$, then go to the right child. Once we commit to a direction, every descendant on the other direction is set to $(3,z)$ for the R nodes and 2 for the A and B nodes.

First assume that $\text{DISJ}(x,y) = 0$. Then $x_{i,z} = y_{i,z} = 1$ for some $z \in [k], i \in [n]$. By Lemma 1 we infer that A_i and B_i are not independent conditioned on either $R_i^1 = (1,z)$ or $R_i^1 = (2,z)$ and hence, $\mathcal{E}_p(G) \neq 0$.

Conversely, assume that $\text{DISJ}(x,y) = 1$. The Local Markov Property says that if every vertex is independent of its non-descendants given its parents then $\mathcal{E}_p(G) = 0$.

- First we show that it is true for any R_i^j variable. Conditioned on the parent of R_i^j taking the value $(3,z)$, R_i^j is constant and hence independent of non-descendants. Conditioned on the parent of R_i^j taking the value $(1,z)$ or $(2,z)$, the values of the non-descendants of R_i^j are fixed and hence independent of R_i^j.
- Next, we show that it is true for any A_i variable. The argument for B_i is identical. Conditioned on $R_i^1 = (3,z)$, then A_i is constant and hence independent of all non-descendants. If $R_i^1 = (1,z)$ or $R_i^1 = (2,z)$, the values of all non-descendants, except possibly B_i, are fixed. But by Lemma 1, B_i is independent of A_i conditioned on R_i^1 since $\text{DISJ}(x,y) = 1$.

Hence, $\text{DISJ}(x,y) = 1$ iff $\mathcal{E}_p(G) = 0$ and therefore testing if $\mathcal{E}_p(G) = 0$ requires $\Omega(nk)$ space.

To extend the lower bound to $\Omega(nk^d)$ consider an instance of DISJ of length nk^d. Let the variables in G be children of all $d-1$ new variables $D_1, \ldots, D_{d-1}$ where there is a directed edge between $D_i \to D_j$ for $i > j$. Call the new network G'. Similar to the proof of Proposition 1, to solve DISJ on the wth pair of bit strings of length nk where $w \in [k^{d-1}]$, Alice and Bob generate samples with variables in G as described above and set $(D_1, \ldots, D_{d-1}) = w$. Hence, any streaming algorithm that decides if $\mathcal{E}_p(G') = 0$ requires $\Omega(nk^d)$ space.

4 Log-Likelihood and Approximate Chow-Liu Trees

While it is natural to test the networks using ℓ_1 or ℓ_2 distance, it is more convenient to use the log-likelihood to learn the structure of certain types of Bayesian networks. Let $\mathbf{x}^{(j)}$ be the jth sample in the stream. The log-likelihood of G given the data stream is:

$$\mathcal{L}(D, G) = \frac{1}{m} \sum_{i=1}^{m} \log \mathcal{P}_G(\mathbf{x}^{(i)}) = -\sum_{j=1}^{n} H(X_j | \operatorname{Pa}(X_j))$$

By using the entropy estimation algorithm of Chakrabarti et al. [3] to estimate the conditional entropies $H(X_j | \operatorname{Pa}(X_j))$ for each of the $O(k^d)$ possible values of $\operatorname{Pa}(X_j)$, we can approximate $\mathcal{L}(D, G)$ up to a factor $1 + \varepsilon$.

Theorem 6. *There is a single-pass algorithm that returns a $(1 + \varepsilon)$ approximation of $\mathcal{L}(D, G)$ for a given Bayesian network G w.h.p using $\tilde{O}(\varepsilon^{-2} n k^d)$ space.*

We prove that the above algorithm is tight in terms of k and d.

Theorem 7. *There exists a Bayesian network G with such that any single-pass streaming algorithm that outputs a 2-approximation of $\mathcal{L}(D, G)$ requires $\Omega(k^d)$ space.*

Proof. Let $t = 10dk^d \log k$. Consider the network with nodes $\{X_i\}_{i \in [t]}$ that are all children of $\{Y_i\}_{i \in [d]}$. Let $\mathbf{Y} = (Y_1, \ldots, Y_d)$. Then,

$$\mathcal{L}(D, G) = -\sum_{i=1}^{t} H(X_i | \mathbf{Y}) - \sum_{i=1}^{d} H(Y_i).$$

Using ideas from [6], we make the following reduction. Given an instance of DISJ with bit strings a, b of length k^d where we may assume $|\{i : a_i = 1\}| = |\{i : b_i = 1\}| = k^d/4$. If Alice and Bob generate samples from the joint distribution $(X_1, \ldots, X_t, \mathbf{Y})$:

$$S_A = \{(1, \ldots, 1, i) : a_i = 1\}, \quad S_B = \{(2, \ldots, 2, i) : b_i = 1\}.$$

where $i \in [k^d]$ specifies the values for $\mathbf{Y}$. If $\operatorname{DISJ}(a, b) = 1$ then $H(X_i | \mathbf{Y}) = 0$ and furthermore $\sum_{i=1}^{t} H(X_i | \mathbf{Y}) = 0$. If $\operatorname{DISJ}(a, b) = 1$ then $H(X_i | \mathbf{Y}) \geq 4/k^d$ and hence, $\sum_{i=1}^{t} H(X_i | \mathbf{Y}) \geq 40d \log k$. Because $\sum_{i=1}^{d} H(Y_i) \leq d \log k$, a 2-approximation of $\mathcal{L}(D, G)$ distinguishes $\sum_{i=1}^{t} H(X_i | \mathbf{Y}) = 0$ from $\sum_{i=1}^{t} H(X_i | \mathbf{Y}) \geq 40d \log k$.

The famous Chow-Liu tree [4], T_{CL}, is the tree with $d = 1$ that maximizes the log-likelihood. Chow-Liu tree is particularly important as it is the only known closed form structural learning algorithm that is polynomial time. We show that there is a single-pass algorithm that approximates T_{CL}.

Theorem 8. *There is a single-pass algorithm that outputs a rooted tree T such that $\mathcal{L}(D, T) \geq (1 - \varepsilon)\mathcal{L}(D, T_{CL})$ with probability at least $1 - \delta$ in $\tilde{O}(n^2 k \varepsilon^{-2} \log(\delta^{-1}))$ space. The post processing time is $O(n^2)$.*

5 Space-Accuracy trade-offs in Independence Testing

From previous work on independence testing [1, 2, 6], we may assume:

Theorem 9. *There exist single-pass algorithms that computes a $(1 \pm \varepsilon)$-approximation of $\mathcal{E}_p(\emptyset)$ with probability at least $1 - \delta$ and uses*

1. $O((\varepsilon^{-1} \log(mk\delta^{-1}))^{O(n)^n})$ *space for* $p = 1$
2. $O(3^n \varepsilon^{-2}(\log k + \log m) \log \delta^{-1})$ *space for* $p = 2$.

By simply appealing to Theorem 2, we have an interesting trade-off between the space usage and the approximation accuracy when testing n-wise independence using ℓ_1 distance. Specifically, we can have an $O(n)$-approximation of $\mathcal{E}_1(\emptyset)$ but using only $O(\text{poly}(n)\,\text{polylog}(k))$ space compared to the space of doubly-exponential in n in Theorem 9.

Proposition 2. *There is a single-pass algorithm that outputs a $O(n)$-approximation of $\mathcal{E}_1(\emptyset)$ using $O(\text{poly}(n, \varepsilon^{-1}) \cdot \text{polylog}(m, k, \delta^{-1}))$ space.*

We can even achieve a stronger approximation guarantee:

Theorem 10. *For any constant $1 \leq t < n/2$, there is a single-pass algorithm that outputs a $(1 \pm \varepsilon)(n-1)/t$-approximation for $\mathcal{E}_1(\emptyset)$ using $\tilde{O}(\text{poly}(n, \varepsilon^{-1}))$ space.*

We can approximate $\mathcal{E}_2(\emptyset)$ as stated in Theorem 9 up to a factor $(1 \pm \varepsilon)$ using $O(3^n)$ space. However, if we allow the error to be additive, we only need $O(n)$ space.

Theorem 11. *There exists an $O(n^3 \varepsilon^{-2} \log(mk) \log \delta^{-1})$-space single-pass algorithm that outputs $\mathcal{E}_2(\emptyset) \pm \varepsilon$ with probability at least $1 - \delta$.*

Proof. The main idea is to rewrite $\mathcal{E}_2(\emptyset)$ as follows:

$$\mathcal{E}_2(\emptyset) = \sum_{\mathbf{x} \in [k]^n} \mathbb{P}(\mathbf{X} = \mathbf{x})^2 + \prod_{i=1}^{n} \sum_{x_i \in [k]} \mathbb{P}(X_i = x_i)^2 - 2 \sum_{\mathbf{x} \in [k]^n} \mathbb{P}(\mathbf{X} = \mathbf{x}) \prod_{i=1}^{n} \mathbb{P}(X_i = x_i).$$

It is possible to estimate the values of $\sum_{x_i \in [k]} \mathbb{P}(X_i = x_i)^2$ for all $i \in [n]$ and $\sum_{\mathbf{x} \in [k]^n} \mathbb{P}(\mathbf{X} = \mathbf{x})^2$ up to a multiplicative factor of $(1 + \varepsilon/n)$ in $O(n^3 \varepsilon^{-2} \log(km + \delta^{-1}))$ space using an existing algorithm for estimating the second frequency moment [10]. This implies a $(1 + \varepsilon)$ multiplicative approximation for the first two terms. However, since $\sum_{\mathbf{x} \in [k]^n} \mathbb{P}(\mathbf{X} = \mathbf{x})^2 \leq 1$ and $\prod_{i=1}^{n} \sum_{x_i \in [k]} \mathbb{P}(X_i = x_i)^2 \leq 1$ this implies an additive 2ε approximation to the first two terms.

It remains to show we can approximate $\sum_{\mathbf{x} \in [k]^n} \mathbb{P}(\mathbf{X} = \mathbf{x}) \prod_{i=1}^{n} \mathbb{P}(X_i = x_i)$ in small space. To argue this let

$$H = \{(x_1, \ldots, x_n) \in [k]^n : \mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) \geq \varepsilon)\}\,.$$

We will show that it is possible to construct a set H' such that $H \subseteq H'$ and for all $(x_1, \ldots, x_n) \in H'$, we may estimate $\mathbb{P}(X_i = x_i)$ and $\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n)$ up to a factor $(1 + \varepsilon)$.

To do this we use the Count-Min sketch [5] which has the following properties:

Claim. There exists a $O(\varepsilon^{-2} \log \delta^{-1}(\log m + \log t))$-space streaming algorithm that, when run on any stream of length m defining a frequency vector y of length t, returns a set of indices and estimates $C = \{(i, \tilde{y}_i) : y_i \leq \tilde{y}_i \leq (1+\varepsilon)y_i\}$ such that $(i, \tilde{y}_i) \in C$ for all $y_i \geq \varepsilon |y|$. We call $S = \{i : (i, \tilde{y}_i) \in C\}$ the ε-cover of y.

In our case y will be a pmf vector, i.e., the frequency vector normalized by dividing each coordinate by m and hence $|y| = 1$. Thus we can find an ε-cover S of the joint pmf and an ε-cover S_i of the marginal pmf of each variable X_i. Let

$$H' = \{(x_1, \ldots, x_n) \in S : x_i \in S_i \text{ for all } i \in [n]\} .$$

Note that if $\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) \geq \varepsilon$ then $\mathbb{P}(X_1 = x_1) \geq \varepsilon, \ldots, \mathbb{P}(X_n = x_n) \geq \varepsilon$. Therefore, the ε-covers constructed using the Count-Min sketch give a multiplicative estimate $\sum_{\mathbf{x} \in H'} \mathbb{P}(\mathbf{X} = \mathbf{x}) \prod_{i=1}^{n} \mathbb{P}(X_i = x_i)$. Furthermore,

$$\begin{aligned} \sum_{\mathbf{x} \notin H'} \mathbb{P}(\mathbf{X} = \mathbf{x}) \prod_{i=1}^{n} \mathbb{P}(X_i = x_i) &\leq \sum_{\mathbf{x} : \mathbb{P}(\mathbf{X} = \mathbf{x}) < \varepsilon} \mathbb{P}(\mathbf{X} = \mathbf{x}) \mathbb{P}(X_1 = x_n) \\ &\leq \varepsilon \sum_{x_1 \in [k]} \mathbb{P}(X_1 = x_1) = \varepsilon. \end{aligned}$$

and therefore the total additive error in our estimate of $\mathcal{E}_2(\emptyset)$ is $O(\varepsilon)$.

References

1. Braverman, V., Chung, K.M., Liu, Z., Mitzenmacher, M., Ostrovsky, R.: Ams without 4-wise independence on product domains. In: STACS. pp. 119–130 (2010)
2. Braverman, V., Ostrovsky, R.: Measuring independence of datasets. In: STOC. pp. 271–280 (2010)
3. Chakrabarti, A., Cormode, G., McGregor, A.: A near-optimal algorithm for estimating the entropy of a stream. ACM Transactions on Algorithms 6(3) (2010)
4. Chow, C., Liu, C.: Approximating discrete probability distributions with dependence trees. IEEE Trans. Inf. Theor. 14(3), 462–467 (Sep 2006), http://dx.doi.org/10.1109/TIT.1968.1054142
5. Cormode, G., Muthukrishnan, S.: An improved data stream summary: the count-min sketch and its applications. Journal of Algorithms 55(1), 58–75 (2005)
6. Indyk, P., McGregor, A.: Declaring independence via the sketching of sketches. In: SODA. pp. 737–745 (2008)
7. Jensen, F.V., Nielsen, T.D.: Bayesian networks and decision graphs. Springer (2007)
8. Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K.: An introduction to variational methods for graphical models. Springer (1998)
9. Kalyanasundaram, B., Schintger, G.: The probabilistic communication complexity of set intersection. SIAM Journal on Discrete Mathematics 5(4), 545–557 (1992)
10. Kane, D.M., Nelson, J., Porat, E., Woodruff, D.P.: Fast moment estimation in data streams in optimal space. In: STOC. pp. 745–754 (2011)
11. Pappas, A., Gillies, D.F.: A new measure for the accuracy of a bayesian network. In: MICAI 2002: Advances in Artificial Intelligence, pp. 411–419. Springer (2002)