On the Complexity of Inference in Probabilistic Graphical Models on Typical-Case Observations

George Matheos, May 6, 2024

1 Background

1.1 Probabilistic graphical models and inference problems

Definition 1. A **probabilistic graphical model on binary variables** is a tuple (V, E, P), where V is an ordered, finite set of variables $V = \{v_1, \ldots, v_n\}$, E is a set of directed edges between the variables, and P is a *conditional probability table*. The directed graph (V, E) must be acyclic. For $v \in V$, $\mathrm{Pa}(v)$ denotes the set of parent variables of v: $\mathrm{Pa}(v) = \{u : (u \mapsto v) \in E\}$. Given any assignment $a_{\mathrm{Pa}(v)} \in \{0,1\}^{|\mathrm{Pa}(v)|}$ to the parent variables of v, the conditional probability table P stores value $P(v = \cdot; a_{\mathrm{Pa}(v)})$, which is a probability vector $[p_{v=0}, p_{v=1}]$ in $\mathbb{R}^2$.

A general probabilistic graphical model lifts the restriction that each variable v_i is binary, and allows it to have arbitrary finite domain. In this report, I will focus on binary probabilistic graphical models. Because a variable with a domain of size k can be represented using $\log k$ binary variables, all the results in this case carry to the general case, except those which restrict the sizes of sets of variables under consideration. Henceforth, the phrase "probabilistic graphical model" should be understood as a graphical model on binary variables.

Given a graphical model (P, E, V), we can define a joint distribution on all the variables in V, with probability mass function

$$P(a) = \prod_{v_i \in V} P(v_i = a_i; a_{\mathrm{Pa}(v_i)}) \quad \forall a \in \{0,1\}^{|V|}$$

where $a_{\mathrm{Pa}(v_i)}$ is the assignment to the parent variables of v_i in a. I will often write P to refer to the whole graphical model, the joint distribution on all its variables, and also marginal and conditional distributions on subsets of its variables.

Definition 2. An **inference problem** consists of a graphical model (V, E, P), a set of *observed variables* $Y \subseteq V$, a set of *query variables* $X \subseteq V$, and an assignment $y \in \{0,1\}^{|Y|}$ to the observed variables such that $P(Y = y) > 0$.

The goal of an inference problem is to compute some piece of information about the posterior distribution $P(X = \cdot | Y = y)$, which is a probability distribution on $\{0,1\}^{|X|}$.

Definition 3. An **inference problem schema** is the tuple $I = (V, E, P, X, Y)$ as in an inference problem, but not fixing an assignment y to the observed variables.

1.2 Worst and typical case inference algorithms

Definition 4. A **deterministic, worst-case additive PDF approximation algorithm with tolerance** ϵ is a Turing machine A which on input (I, y, x), where I is any inference problem schema, y is any assignment to the observed variables, and x is an assignment to the query variables, outputs a rational number $A(I, y, x)$ such that

$$|A(I, y, x) - P(X = x | Y = y)| < \epsilon$$

In 1993, Dagum and Luby [1] showed that if there exists a worst-case additive PDF approximation algorithm with tolerance $< 1/2$, and it has polynomial runtime, then $\mathbf{P} = \mathbf{NP}$.

Probabilistic graphical models are typically used to model aspects of the world. That is, each variable in V represents some aspect of the world; Y represents a set of values which we have observed; and X represents a set of values which we wish to infer. In this setting, the probability distribution P is a description of our

beliefs about how probable different joint outcomes of events in the world are. Therefore, given a graphical model P in which we must do inference, it may be acceptable to us if there is exist assignments y under which computing the posterior distribution is very expensive, so long as these instances are extremeley rare. Since we have a probability distribution P on hand which ought to roughly correspond to the distribution of y values which will occur in the world, and on which we will have to run inference, a natural notion of "rare" is available. Say there is a small set of observation assignments $\mathcal{Y}_{\text{hard}} \subseteq \{0,1\}^{|Y|}$ such that $P(\mathcal{Y}_{\text{hard}}) < \rho$ for very small ρ (e.g. $\rho = 0.00001$), such that for all $y \notin \mathcal{Y}_{\text{hard}}$, we can compute the posterior distribution $P(X = \cdot|Y = y)$ efficiently. Then we can say that inference is easy in the typical case, and for many purposes this is sufficient.

In fact, Dagum and Luby's construction of a worst-case inference problem I involves selecting an observed assignment y to a set Y of one variable ($|Y| = 1$) which has extremely low marginal probability: $P(Y = y) << 1$. Theorem **??** later in this report shows that for every inference problem schema (P, V, E, X, Y) with $|Y| = 1$, efficient inference is possible on typical case observations. This indicates that Dagum and Luby's strategy for proving the hardness of inference in the worst case does not directly carry through to showing the hardness of inference with typical-case observations.

Definition 5. A **deterministic, typical-case additive PDF approximation algorithm with tolerances** (ϵ, ρ) is a Turing machine A which accepts inputs of the form (I, y, x), where I is any inference problem schema, y is any assignment to the observed variables, and x is an assignment to the query variables, and outputs a rational number $A(I, y, x)$ with the following property. For any graphical model (V, E, P) and any subset $Y \subseteq V$, there exists a "typical set" $\mathcal{Y}_{\text{easy}} \subseteq Y$ such that

$$P(Y \in \mathcal{Y}_{\text{easy}}) > 1 - \rho$$

and for all $X \subseteq V$, any assignment x to X, any extension $Y' \subseteq V$ to Y, so $Y \subseteq Y'$, and any $y' \in \{0,1\}^{|Y|}$ such that $y'_Y \in \mathcal{Y}_{\text{easy}}$,

$$|A(I, y, x) - P(X = x|Y = y)| < \epsilon$$

That is, on all but a small fraction ρ of y values, the algorithm can outputs an ϵ-approximation to the posterior PDF value $P(X = x|Y' = y')$, for any query variables X, any query assignment x, and any observation constraint $Y' = y'$ which contains within it that $Y = y$.

The preceding definition implies that for any $I = (P, E, V, X, Y)$ and any x,

$$\Pr_{y \sim P(Y=\cdot)}[|A(I, y, x) - P(X = x|Y = y)| \geq \epsilon] < \rho$$

The condition in definition 5 is a bit stronger than this probability bound, as the condition in the definition requires that for any typical observation y, it is easy to condition not only on y, but also on any assignment y' extending y. That is, for any $y \in \mathcal{Y}_{\text{easy}}$, $P(V|Y = y)$ is a probability distribution in which both efficient marginal probability computation and efficient conditional probability computation is possible: for any $X \subseteq V$ we can approximate $P(X|Y = y)$, and for any $X, Z \subseteq V$, we can approximate $P(X|Y = y, Z = z)$.

There are also natural versions of these definitions admitting the use of probabilistic algorithms, rather than deterministic ones. [TODO.]

1.3 One way functions

Proving the hardness of a computational problem C is often done by reducing from an **NP**-hard problem like 3SAT to problem C, thereby showing that if C could be solved in polynomial time, $\mathbf{P} = \mathbf{NP}$. However, problems like 3SAT are stated in terms of behavior on all, and thus worst-case, inputs. Therefore, to prove the hardness of a computational problem on typical-case inputs, it is preferable to reduce to a hardness conjecture stated directly in terms of typical-case behavior. One such conjecture is the existance of *one-way functions*, functions which can be efficiently computed, but not efficiently inverted for the majority of inputs. It is widely believed that such functions exist [], as there are a number of functions like multiplication of prime numbers, for which no inversion algorithms are known which are efficient in the typical case. Such functions are widely used in public-key cryptography.

Definition 6. A **one-way function** f is a function $f : \{0,1\}^* \to \{0,1\}^*$ such that f can be computed in polynomial time, and for every probabilistic polynomial time algorithm A,

$$\Pr_{x \sim \text{Uniform}(\{0,1\}^n)}[A(f(x), 1^n) \in f^{-1}(f(x))] \underset{n \to \infty}{\to} 0$$

where $f^{-1}(f(x))$ is the set $\{x' \in \{0,1\}^* : f(x') = f(x)\}$.

2 Hardness results for typical-case inference

In this section, I state a couple of basic (though so far as I can tell, new) results indicating that it is unlikely a polynomial-time algorithm can approximate the posterior distribution on typical-case observations.

Theorem 1. Say that there exists a polynomial-time, deterministic, typical-case additive PDF approximation algorithm with tolerances $\epsilon < 1/2, \rho < 1$. Then one-way functions do not exist.

Proof. For contradiction, say that there exists a one-way function f, computed by Turing maching F, and there exists a deterministic typical-case additive PDF approximation algorithm A with tolerances $\epsilon < 1/2, \rho < 1$. I will show that this implies the existance of a polynomial time algorithm B which inverts f on many inputs, contradicting that f is a one-way function.

By Lemma 2, without loss of generality, we can assume that if $|f(x)| = s$ for any $x \in \{0,1\}^n$, then $|f(x')| = s$ for all $x' \in \{0,1\}^n$.

Algorithm B will behave as follows. On input $(y, 1^n)$ the goal of B is to output a value $x \in \{0,1\}^n$ such that $f(x) = y$. Given this input, B first constructs the description of the following inference problem schema (V, E, P, X, Y).

Let C be a circuit on n inputs with $|y|$ outputs, of size polynomial in n, which computes f on all inputs of length n. (Our Turing machine B can write down a description of C for any n, in time polynomial in n, using the Cook-Levin reduction.) Let $G = \{g_1, \ldots, g_{|}G|\}$ be an ordering of the set of gates in circuit C. Let $Y \subseteq G$ be the set $|y|$ gates that feed to the output lines of the circuit. Let $X = \{x_1, \ldots, x_n\}$ be an ordered set of n "input variables". Let (P, E, V) be a probabilistic graphical model with $V = G \cup X$, which implements the following logic: (1) sample a string x uniformly from $\{0,1\}^n$; (2) compute $f(x)$ using the circuit C; and then (3) output the value $y = f(x)$ computed by the circuit. In slightly more detail, construct E to contain all edges between the gates in C, and to contain edges from x_i to gate g_j for every gate g_j receiving input from input bit i. Construct P so that for each $g \in G$, $P(g|a_{\text{Pa}}(g))$ is a deterministic probability table that computes the operation of gate g, and for each $x_i \in X$, $P(x_i; \emptyset) = [\frac{1}{2}, \frac{1}{2}]$. This yields an inference problem schema (V, E, P, X, Y). (For more details on the construction, see Appendix **??**).

Algorithm B then proceeds as follows. Let $X_1 = \{x_1\}$, where $x_1 \in X$. Observe that (V, E, P, X_1, Y) is an inference problem schema. Algorithm B runs algorithm A on input $(V, E, P, X_1, Y, y, 1)$, and computes a value p_1 such that, if y is a typical-case value

$$|P(x_1 = 1|Y = y) - p_1| < \epsilon$$

(The details regarding y being a typical case value will be elaborated on below.) If $p_1 > \frac{1}{2}$, set $x_1 = 1$. Otherwise, set $x_1 = 0$.

Next, B constructs a new inference problem schema (V, E, P, X', Y') where $X' = X \setminus \{x_1\}$ and $Y' = Y \cup \{x_1\}$. Set $X_2 = \{x_2\}$, and use algorithm A to compute p_2 approximating $P(X_2 = 1|Y = y, X_1 = x_1)$. Set $x_2 = 1$ if $p_2 > 1/2$. Proceed in this manner, to generate an assignment $(x_1, \ldots, x_n)$. Then, output $(x_1, \ldots, x_n)$.

It turns out (to be proven momentarily) that

$$\forall n, \Pr_{x \sim \text{Uniform}(\{0,1\}^n)}[f(B(f(x))) = f(x)] \geq 1 - \rho$$

Because $\rho < 1$, this implies that this probability term cannot converge to 0 as n increases. Thus f is not a one-way function, contradicting the setup.

What remains is to show that the value $(x_1, \ldots, x_n)$ returned by algorithm B satisfies $f(x_1 \ldots x_n) = y$ with probability $\geq 1 - \rho$, if $y = f(x)$ for an x chosen uniformly at random from $\{0,1\}^n$. Observe that this distribution over values y is identical to the distribution $P(Y = y)$ induced by the probabilistic graphical model above.

By our definition of A, there exists a set $\mathcal{Y}_{\text{easy}}$ such that $P(Y \in \mathcal{Y}_{\text{easy}}) > 1 - \rho$. Due to this probability bound, it suffices to prove that algorithm B outputs $(x_1, \ldots, x_n)$ with $f(x_1, \ldots, x_n) = y$ for any $y \in \mathcal{Y}_{\text{easy}}$. Thus, henceforth, assume $y \in \mathcal{Y}_{\text{easy}}$.

If it were the case that for all $x' \in \{0,1\}^{n-1}$, $f(1 \cdot x') \neq y$, then $P(x_1 = 1 | Y = y) = 0$. Thus, if $p_1 > 1/2$, we must have $P(x_1 = 1 | Y = y) > 0$, so there exists x' such that $f(1 \cdot x') = y$. Otherwise, it must be that $P(x_1 = 0 | Y = y) > 0$, so there exists x' such that $f(0 \cdot x') = 0$. Either way, algorithm B chooses a value for x_1 which can be extended into a value in $f^{-1}(y)$.

Now, say that we have an assignment $(x_1, \ldots, x_i)$ such that there exists $x' \in \{0,1\}^{n-i}$ so $f(x_1, \ldots, x_i \cdot x') = y$. Then $P(Y = y, X_{1:i} = x_{1:i}) > 0$, the value $P(X_{i+1} = 1 | Y = y, X_{1:i} = x_{1:i})$ is well-defined. By our definition of A, algorithm A can compute an additive ϵ approximation p_{i+1} of this value, as the assignment being conditioned on, $Y \cup X_{1:i}$, is an extension to assignment y. If $p_{i+1} > 1/2$, then it must be the case that $P(X_{i+1} = 1 | Y = y, X_{1:i} = x_{1:i}) > 0$, implying that $(x_1, \ldots, x_i, 1)$ can be extended into an assignment x such that $f(x) = y$. Otherwise, $p_{i+1} < 1/2$, so by symmetric logic, $(x_1, \ldots, x_i, 0)$ can be extended to a value in $f^{-1}(y)$.

This induction shows that if $y \in \mathcal{Y}_{\text{easy}}$, algorithm B is guaranteed to find an assignment $(x_1 \ldots, x_n)$ so $f(x_1, \ldots, x_n) = y$.

□

Lemma 2. Say that there exists a one-way function f. Then there exists a one-way function g such that the output length is a function of the input length. That is, there exists a one-way function g, and a function $s : \mathbb{N} \to \mathbb{N}$ such that $\forall n, \forall x \in \{0,1\}^n$, $|g(x)| = s(n)$.

Proof. TODO □

3 Attempt at reducing from PRGs

Let $G : \{0,1\}^* \to \{0,1\}^*$ be a pseudorandom generator of stretch $l(n)$. Let (P_n, V_n, E_n) be a probabilistic graphical model which (1) samples a seed s uniformly at random from $\{0,1\}^n$; (2) samples a value a uniformly at random from $\{0,1\}^{l(n)}$; (3) samples a choice bit x uniformly at random from $\{0,1\}$, (4) computes $b \leftarrow G(s)$; and (5) outputs a if $x = 0$ and b if $x = 1$. Let Y denote the set of $l(n)$ output bits $y_1 \dots y_{l(n)}$. Let X be the set containing only x. Given a string y, $P(x|Y = y)$ asks how likely it was that y came from the PRG, or is truly uniform in $\{0,1\}^{l(n)}$. Note that

$$y \notin G(\{0,1\}^n) \implies P(x = 1|Y = y) = 0$$

and

$$y \in G(\{0,1\}^n) \implies P(x = 1|Y = y) = \frac{(m_y)/2^n}{m_y/2^n + 1/2^{l(n)}}$$

where m_y is the number of distinct values in $\{0,1\}^n$ which get sent to y by G. Evaluating the derivative $dP(x = 1|Y = y)/dm_y$ shows it is positive for all $m_y \geq 1$, so the minimum value of this function on this region occurs when $m_y = 1$:

$$y \in G(\{0,1\}^n) \implies P(x = 1|Y = y) \geq \frac{2^{-n}}{2^{-n} + 2^{-l(n)}} = \frac{2^{l(n)-n}}{1 + 2^{l(n)-n}}$$

so

$$y \in G(\{0,1\}^n) \implies P(x = 0|Y = y) \leq \frac{1}{1 + 2^{l(n)-n}} < \frac{1}{2^{l(n)-n}}$$

So if we have nontrivial stretch, we get a super high posterior probability that the string came from the PRG.

Observe that if we had an ϵ approximation for $\epsilon < \frac{1}{2} - 2^{n-l(n)}$, we could distinguish the two cases.

Let ϵ obey this bound. Say we have an algorithm A which on $4/5$ of y values sampled from P computes an ϵ additive approximation to $P(x = 1|y)$. Let $\mathcal{Y}$ be this set of y values. Let B be an algorithm which on input y outputs 1 if the approximation to $P(x = 1|y)$ is greater than $1/2$ and outputs 0 otherwise.

Our construction establishes that

$$P(\mathcal{Y}) = \frac{1}{2} \Pr_{x \sim \{0,1\}^n}[G(x) \in \mathcal{Y}] + \frac{1}{2} \Pr_{a \sim \{0,1\}^{l(n)}}[a \in \mathcal{Y}]$$

Thus, since $P(\mathcal{Y}) \geq 4/5$, we have that either

$$\Pr_{x \sim \{0,1\}^n}[G(x) \in \mathcal{Y}] \geq 4/5 \quad \text{(case 1)}$$

or

$$\Pr_{a \sim \{0,1\}^{l(n)}}[a \in \mathcal{Y}] \geq 4/5 \quad \text{(case 2)}$$

(or both). Further, in either case, it is guaranteed that

$$1/2 \Pr_{x \sim \{0,1\}^n}[G(x) \in \mathcal{Y}] \geq 3/5 \text{ and } \Pr_{a \sim \{0,1\}^{l(n)}}[a \in \mathcal{Y}] \geq 3/5$$

This follows because for either of these probability values p we must have $\frac{1}{2} + \frac{1}{2}p \geq 4/5$ and thus $\frac{1}{2}p \geq \frac{3}{10} \implies p \geq 3/5$.

Thus in case 1,

$$\Pr_{x \sim \{0,1\}^n}[B(G(x)) = 1] \geq 4/5$$

and

$$\Pr_{a \sim \{0,1\}^{l(n)}}[B(a) = 1] \leq 2/5$$

so

$$|\Pr_{x\sim\{0,1\}^n}[B(G(x))=1]-\Pr_{a\sim\{0,1\}^{l(n)}}[B(a)=1]|\geq 2/5$$

In case 2,

$$\Pr_{x\sim\{0,1\}^n}[B(G(x))=1]\geq 3/5$$

and

$$\Pr_{a\sim\{0,1\}^{l(n)}}[B(a)=1]\leq 1/5$$

so

$$|\Pr_{x\sim\{0,1\}^n}[B(G(x))=1]-\Pr_{a\sim\{0,1\}^{l(n)}}[B(a)=1]|\geq 2/5$$

Thus if B is a polynomial time algorithm, G cannot be a 1/6 PRG.

3.1 Probabilistic algorithms

The preceding section shows that if secure PRGs exist then there cannot be a ptime deterministic algorithm which can do inference in any graphical model up to additive error on 4/5-typical observations. A next question is whether there exist probabilistic algorithms which do this.

Say there exist secure PRGs with exponential stretch (that is, with $O(\log(l))$ seeds). Say we have a probabilistic ptime algorithm such that for any graphical model (P,E,V), there is a set $\mathcal{Y}$ with $P(\mathcal{Y})\geq 4/5$ s.t.

$$\forall y\in\mathcal{Y},\Pr_r[|A((P,E,V),X,Y,x,y)-P(X=x|Y=y)|>\epsilon]<1/3$$

Consider how algorithm A would be used by our algorithm B in the preceding section. Algorithm B would sometimes check if the output of A is $\geq 1/2$. If so, our probbilistic A would return a value $\geq 1/2$ w.p. $>2/3$. But we can use our exponential stretch PRG to check if $P(A(...)>1/2)>2/3$ deterministically.

This shows that if exponential-stretch PRGs exist, then no probabilistic algorithm can do typical-case inference.

3.2 Some notes

1. Is it the case that for any probabilistic graphical model, there is a ptime TM that does typical-case inference? The above proof shows that if so, PRGs don't exist.

2. Is it the case there is a ptime TM that, given a graphical model as input, does typical-case inference? This existance is strictly stronger than the above existence, and therefore also implies that PRGs don't exist.

3. If the exponential stretch thing is too strong, I can also use my reduction from 1-way functions to show that relative approximation is hard. (And this might be good to write up, since I have it anyway.)

4 Misc

4.1 The worst-case complexity of inference

On worst and typical case complexity. Cooper [2] and Dagum and Luby [1] have proven that if there exists an $\epsilon < 1/2$ and a polynomial time algorithm A such that for any inference problem (I, y), $|A(I, y, x) - P(X = x|Y = y)| < \epsilon$, then $\mathbf{P} = \mathbf{NP}$. Their constructions involve constructing inference problems I, and then selecting an observed assignment y to a set Y of one variable ($|Y| = 1$). This assignment may have extremely low marginal probability: $P(Y = y) << 1$. As probabilistic graphical models are typically used to model aspects of the world, we ought expect that in practice, the observation values which we must condition our inferences upon will not be worst-case values, but typical-case values. Thus, in this project, I propose to study the complexity of inference under arbitrary (worst case) graphical models (P, V, E), arbitrary latent variable subsets X, and arbitrary observation variable subsets Y, but typical-case observation values y.

My plan is to prove the following results suggesting that typical-case inference can be difficult.

1. Say that there exists $\epsilon, \delta < 1/2$ and a polynomial-time algorithm A such that for any I and any assignment x,
$$\Pr_{y \sim P}[|A(I, y, X) - P(X = x|Y = y)| > \epsilon] < 1 - \delta$$
Then one-way functions do not exist.

2. The above result also holds if A is a probabilistic rather than deterministic algorithm. (The proof will involve a Chernoff style bound.)

3. Corollary: if one-way functions exist, then approximate posterior sampling given typical case observations, can be hard. There does not exist a probabilistic polynomial time algorithm A where $A(I, y)$ outputs an assignment x, such that its output distribution O has bounded total variation distance from the posterior: for any $\epsilon < 1/2$, we cannot have $||O - P(X|Y = y)||_{\text{TV}} < \epsilon$ for all inference problems.

These results will use similar constructions to Dagum and Luby's [1], though some extensions and modifications are needed to reduce from one-way functions rather than from SAT solving.

My plan is to additionally prove the following results about several cases in which typical-case inference *is* possible.

1. Say $|Y| = 1$. Then algorithms A as in item (2) and (3) in the previous list exists. (This is of interest because it shows that the constructions in [2] and [1] do not directly carry through to showing the hardness of inference with typical-case observations.)

2. Say we have either an upper bound B_I on the mutual information $I(X; Y)$, or an upper bound B_χ on the χ^2-information $I_{\chi^2}(X; Y)$ [3]. Then probabilistic polynomial time algorithms A as in items (2) and (3) on the previous list exist, which work for all graphical models satisfying these upper bounds. (That is, the parametrized complexity of inference, on typical-case observations, with either of these information quantities as parameters, is polynomial.) Given a χ^2-information upper bound B_χ, we can bound the algorithm runtime by a polynomial in B_χ. (We cannot get a polynomial bound based on the Shannon mutual information bound.)

3. (Tentative.) Inference in graphical models with upper-bounded mutual information $I(X; Y)$ on worst-case observations cannot be done in polynomial time unless $\mathbf{P} = \mathbf{NP}$. (The ability to state bounds like this therefore requires typical-case analysis.)

These upper bounds on the complexity of information in information-limited settings

I am planning to also include a discussion briefly surveying all of the papers about compmlexity of inference which I read during this project, to give you a sense of what I have studied (much of which ended up not being directly informative about these results).

5 Papers I have read

Here is a list of most of the papers I have read as a part of this course project: [4, 2, 1, 5, 6, 7, 8, 9, 10]. I also looked at [11, 12], but I have not read them in detail. My plan at this stage is to halt my literature review and try to write a project report proving the statements I described above.

I really enjoyed the papers about Majority K-SAT, by the way! I watched Shyan's video about it in addition to reading the paper, and thought it did a great job making clear the structural decomposition of the space of CNF formulae that informs the result. I was not able to figure out any new results related to this I thought I could prove, so for this project I thought I would write a report on the topics outlined above. But I'm quite interested in the line of research studying the combinatorial structure in concrete classes of probabilistic models, and understanding how this affects the complexity of probabilistic operations in them. (In the k-sat work, the probabilistic class of models is "sample n binary variables uniformly at random, then compute these CNF formulae on them", and the probabilistic operation under study is to compute a threshold on the marginal probability of observing all the CNF formlae.)

References

[1] Paul Dagum and Michael Luby. "Approximating probabilistic inference in Bayesian belief networks is NP-hard". In: *Artificial intelligence* 60.1 (1993), pp. 141–153.

[2] Gregory F Cooper. "The computational complexity of probabilistic inference using Bayesian belief networks". In: *Artificial intelligence* 42.2-3 (1990), pp. 393–405.

[3] Yury Polyanskiy. *f-divergences*. Lecture notes. URL: `https://people.lids.mit.edu/yp/homepage/data/LN_fdiv.pdf`.

[4] Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. "Random generation of combinatorial structures from a uniform distribution". In: *Theoretical computer science* 43 (1986), pp. 169–188.

[5] Paul Dagum and Michael Luby. "An optimal approximation algorithm for Bayesian inference". In: *Artificial Intelligence* 93.1-2 (1997), pp. 1–27.

[6] Nathanael L Ackerman, Cameron E Freer, and Daniel M Roy. "On the computability of conditional probability". In: *Journal of the ACM (JACM)* 66.3 (2019), pp. 1–40.

[7] Johan Kwisthout. "Approximate inference in Bayesian networks: Parameterized complexity results". In: *International Journal of Approximate Reasoning* 93 (2018), pp. 119–131.

[8] Shyan Akmal and Ryan Williams. "majority-3sat (and related problems) in polynomial time". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 1033–1043.

[9] Till Tantau. "On the Satisfaction Probability of k-CNF Formulas". In: *arXiv preprint arXiv:2201.08895* (2022).

[10] Scott Aaronson. "The equivalence of sampling and searching". In: *Theory of Computing Systems* 55.2 (2014), pp. 281–298.

[11] Uriel Feige. "Relations between average case complexity and approximation complexity". In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. 2002, pp. 534–543.

[12] Ankur Moitra. "Approximate counting, the Lovász local lemma, and inference in graphical models". In: *Journal of the ACM (JACM)* 66.2 (2019), pp. 1–25.