On the Complexity of Statistical Reasoning*

Joe Kilian[†] Moni Naor [‡]

Revised August 23, 1998

Abstract

We show that basic problems in reasoning about statistics are NP-hard to even approximately solve. We consider the problem of detecting internal inconsistencies in a set of statistics. We say that a set of statistics is ϵ-inconsistent if one of the probabilities must be off by at least ϵ. For a positive constant ϵ, We show that it is NP-hard to distinguish ϵ-inconsistent statistics from self-consistent statistics. This result holds when restricted to complete sets of pairwise statistics over Boolean domains.

We next consider what may be determined about distributions with a given (consistent) set of pairwise statistics over Boolean domains. We show that it is NP-hard to distinguish between the case that $\Pr[X_i \wedge X_j]$ is necessarily 0 and the case that $\Pr[X_i \wedge X_j]$ can have any value in $[0, \frac{1}{2}]$. Similarly, we show it NP-hard to distinguish between the case that $|Corr(X_i, X_j)| = -1$ and the case that $|Corr(X_i, X_j)|$ is unconstrained.

Whereas the connection between PCP and hardness of approximations has been known since Feige et. al [9], we introduce the application of "zero-knowledge" PCP's as a tool for proving NP-hardness results for approximation problems.

*Preliminary version appeared in Proc. 3rd Israeli Symp. on Theory of Computing and Systems, 1995, pp. 209–217.

†NEC Research Institute, 4 Independence Way, Princeton, New Jersey, USA. E-mail: joe@research.nj.nec.com.

‡Incumbent of the Morris and Rose Goldman Career Development Chair, Dept. of Applied Mathematics and Computer Science, Weizmann Institute, Rehovot 76100, Israel. E-mail: naor@wisdom.weizmann.ac.il. part of this work was done while at IBM Almaden Research Center. Research supported by an Alon Fellowship and by a grant from the Israel Science Foundation administered by the Israeli Academy of Sciences.

1 Introduction.

1.1 Consistency checking and estimating probabilities

Much of reasoning consists of gauging the plausibility of ones beliefs and using these beliefs to infer the possibility or likelihood of other events. We consider basic problems in reasoning about assertions of the form "the probability of event E is p," or more compactly, $\Pr[E] = p$. Given a set S of probabilistic constraints, two natural questions one can ask are

- Is S internally consistent?
- Based on S, what can be said about the probability of some event E whose probability may not be explicitly given by S?

For example, the constraint set $\{\Pr[X_1] = .8, \Pr[\neg X_1] = .3\}$ is inconsistent. The set $\{\Pr[X_1] = .8, Pr[X_2] = .7\}$ is consistent, and allows us to bound the probability of $X_1 \wedge X_2$ by

$$.5 \leq \Pr[X_1 \wedge X_2] \leq .7.$$

No stronger statement is possible, since for any $y \in [.5, .7]$ there exists a probability measure in which $\Pr[X_1] = .8, \Pr[X_2] = .7$ and $\Pr[X_1 \wedge X_2] = y$.

We show that the above problems, and even natural approximate versions of these problems, are NP-hard to solve under a very simple set of constraints.

1.2 Previous work

If ones constraints are sufficiently general, then one can trivially code up 3SAT problems in this framework. For instance, given a SAT expression $S = C_1 \wedge \cdots \wedge C_m$, the set of constraints $\{\Pr[C_i] = 1\}$ is consistent if and only if S is satisfiable. Koller and Megiddo [14] (see also [8]) consider sets of *k-simple constraints*, which are of the form

$$\Pr\left[X_{i_1} = b_1 \wedge \cdots \wedge X_{i_k} = b_k\right] = y,$$

where each variable X_i ranges over some fixed domain. They prove the following theorem, which is a point of departure for one of our results.

Proposition 1 [Koller-Megiddo] Given a set S of 2-simple constraints over Boolean variables $X_1, \ldots, X_n$, it is NP-hard to determine if S is consistent.

We extend this result in a number of ways, proving an NP-hardness result under a somewhat more restricted set of constraints and allowing for a much weaker notion of consistency checking.

There has also been a great deal of work on the complexity of counting solutions to a given formula. For example, Valiant [19, 20] considers the problem of counting the number of solutions to 2-variable monotone CNF formulas, and shows it to be $\#P$-complete. Roth [18] shows $\#P$-hardness results for counting solutions of many other restricted classes of formula. While there is some connections between counting and computing probabilities, we know of know direct connections between these results and ours.

An area in statistical reasoning which has been investigated from computational complexity point of is Bayesian belief networks (see [4, 16, 17] for information about them). Cooper [6] and Dagum and Luby [7] have shown that is is hard to even approximate the probabilities in such networks.

1.3 Approximate Consistency, Probability Intervals and Implied Correlations

The Koller-Megiddo result considers the standard, qualitative notion of consistency: either a set of constraints is consistent or it is not. However, it is useful to quantify the notion of consistency, since there are many possibilities for introducing errors in the probabilities of the constraints. For example, the statement that 66% of people don't trust statistics and 33% do is inconsistent, but this inconsistency might be due to roundoff errors, and is intuitively less troubling than a 40%/80% split. Another source of errors is due to sampling. For instance, suppose that the constraints were generated by examining a subset of the variables and figuring out the distribution on them via repeated samples. Luckily we have statistical tools that tell us that these errors are small with high probability, However, we cannot require perfect consistency. This motivates us to consider the following quantitative definition of consistency, which measures how closely an inconsistent set of constraints can be satisfied by a probability measure.

We first set up some notation.

Definition 1 Let S be a set of probabilistic constraints. Given an event E, we define $S(E) = p$ if there is a constraint in S of the form $\Pr[E] = p$ and $S(E) = \perp$ if no such constraint exists. We use the bastardized notation $E \in S$ to denote the fact that a constraint on E is given in S.

Definition 2 Let S be a set of probabilistic constraints and let $\mathcal{P}$ be a probability measure. We say that $\mathcal{P}$ $(1-\epsilon)$-*satisfies* S if for all constraints $E \in S$, $|\mathcal{P}(E) - S(E)| \leq \epsilon$. We say that S is $(1-\epsilon)$-consistent if there exists a probability measure that $(1-\epsilon)$-satisfies S and that S is ϵ-*inconsistent* if it is not $(1-\epsilon)$-consistent.

In the above example, a 66%/33% split is .005-inconsistent, since there is a probability measure achieving a 66.5%/33.5% split, whereas an 80%/40% split is .2-inconsistent. The motivation behind our definition is that if a constraint set is ϵ-inconsistent, then at least one the probabilities must be changed by at least ϵ. Note that if we a given a probability measure $\mathcal{P}$ in a black box form (i.e. we can sample from $\mathcal{P}$ and examine the assignment to the variables), then we can distinguish with high probability between the following two cases:

1. $\mathcal{P}$ is ϵ-consistent with a given set of constraints

2. $\mathcal{P}$ is ϵ'-inconsistent

for any $\epsilon' > \epsilon > 0$. The time (which is is determined by the number of samples) is polynomial in n and $\frac{1}{\epsilon' - \epsilon}$.

A possible objection to this quantitative notion of consistency is that it is "worst case." Under this measure, it is more consistent to have a large number of constraints that deviate by .19 than a single constraint that deviates by .2. This motivates a weighted version of ϵ-consistency.

Definition 3 Let S be a set of probabilistic constraints, and let ω assign nonnegative weights, summing to 1, to the events in S. Let $\mathcal{P}$ be a probability measure. We say that $\mathcal{P}$ $(1-\epsilon)$-*satisfies* S weighted by ω if

$$\sum_{E \in S} \omega(E)|\mathcal{P}(E) - S(E)| \leq \epsilon.$$

We say that S is $(1-\epsilon)$-consistent weighted by ω if there exists a probability measure that $(1-\epsilon)$-satisfies S weighted by ω and that S is ϵ-*inconsistent* weighted by ω if it is not $(1-\epsilon)$-consistent weighted by ω.

Our main result for approximate consistency checking is as follows:

Theorem 1 Let S be a set of 2-simple constraints on Boolean variables $X_1, \ldots, X_n$. Then for an absolute constant $\epsilon > 0$, it is NP-hard to distinguish whether S is consistent or S is ϵ-inconsistent. This result remains true even if S is required to assign a probability to every pairwise event.

That is, given a promise that either S is completely consistent, or that S is ϵ-inconsistent, one still cannot efficiently determine which is the case unless $NP = BPP$.

Similarly, we show that the weighted version is also hard.

Theorem 2 Let S be a set of 2-simple constraints on Boolean variables $X_1, \ldots, X_n$, and let ω be a measure on events as defined in Definition 3. Then for an absolute constant $\epsilon > 0$, it is NP-hard to distinguish whether S is consistent or S is ϵ-inconsistent weighted by ω.

Given a set of probabilistic constraints of the form $\Pr[E_i] = p_i$, consider the problem of computing the set of possible probabilities for an event E. By a simple convexity argument the set of possible probabilities for an event E will consist of an interval (possibly of the form $[p,p]$). If $\mathcal{P}_1$ and $\mathcal{P}_2$ satisfy S then for any $\lambda \in [0,1]$ the probability measure $\mathcal{P}' = \lambda\mathcal{P}_1 + (1-\lambda)\mathcal{P}_2$ satisfies S. Furthermore, if $\mathcal{P}_1(E) = x$ and $\mathcal{P}_2(E) = y$ then $\mathcal{P}'(E) = \lambda x + (1-\lambda)y$. We call this interval the *probability interval* for E. Given a probabilistic constraint set S and an event E a natural problem is to compute E's probability interval $[p_1, p_2]$ (for all the cases we will consider the probability interval is closed). A natural approximation problem is to compute p_1' and p_2' such that $|p_i' - p_i| \leq \epsilon$. We say that such an approximation computes the probability interval to within ϵ. Our main result on the complexity of this problem is as follows:

Theorem 3 Given a set S of 2-simple constraints on Boolean $X_1, \ldots, X_n$, it is NP-hard to distinguish whether the probability interval of the event $X_1 \wedge X_2$ is $[0, \frac{1}{2}]$ or $[0,0]$.

Another natural question to ask is what a set of statistics implies about the correlation between two given variables, x_i and x_j. We show it NP-hard to even approximately compute which forms of correlation are allowed or ruled out.

Theorem 4 Let S be a set of 2-simple constraints on Boolean variables $X_1, \ldots, X_n$. It is NP-hard to distinguish whether for variables (X_i, X_j) which of the following cases holds

- $Corr(X_i, X_j)$ is unconstrained.

- $Corr(X_i, X_j) = -1$.

Here, $Corr(X_i, X_j)$ is defined by

$$Corr(X_i, X_j) = \frac{\Pr[X_i X_j] - \Pr[X_i]\Pr[X_j]}{\Pr[X_i]\Pr[X_j]}.$$

1.4 Techniques Used

To prove our main results, we build on recent developments in the theory of probabilistic checkable proofs (PCP's) and zero-knowledge PCP's. Arora et al. [1] show that any proof P of a theorem T can be converted into a PCP P' of size polynomial in $|P|$ and in which the probabilistic polynomial time (PPT) verifier V probes $O(1)$ bits of P'. Furthermore, (P', V) exhibits the following completeness and soundness properties.

(perfect completeness) If P' is correct, then V always accepts.

(soundness) If T is false, then V rejects with probability $\frac{1}{2}$.

For convenience, we say that (P, V) achieves an (α, β) separation if V accepts a correct proof with probability $\geq \beta$ and an incorrect theorem with probability $< \alpha$. Thus, the [1] result achieves a $(\frac{1}{2}, 1)$ separation. The results of Dwork et al. [5] add zero-knowledge properties to PCP's, at the price of a weaker soundness condition. The new properties are as follows.

(weak soundness) The proof achieves a $(1 - \epsilon, 1)$ separation, for some constant ϵ.

(zero-knowledge) One can simulate V's view in probabilistic polynomial time.

To achieve the zero-knowledge property, one must consider "probabilistic PCP's," PCP's which are generated according to some probability measure.

In order to prove the strongest known results on the hardness of approximate consistency checking, we establish a correspondence between zero-knowledge PCP's and distributions satisfying certain systems of constraints. Probing k bits of a proof corresponds to a k-simple constraint, so to obtain the strongest possible qualitative results we minimize the number of bits queried. Dwork et al. [5] mention (but do not describe) a technique by which a proof may be transformed so that the verifier only looks at 3 bits, while still rejecting incorrect theorems with probability $\Omega(1)$. Under their definition of PCP proofs, which requires perfect completeness, this is optimal. However, we show that under a relaxed completeness requirement there are PCP's for NP in which the verifier only probes two bits of the proof.

Theorem 5 Any proof P of a theorem T can be randomly converted into a probabilistic PCP $\mathcal{P}$ of size polynomial in $|P|$, in which V probes 2 bits of $\mathcal{P}$. Furthermore, $(\mathcal{P}, V)$ exhibits the following completeness, soundness and zero-knowledge properties.

weak completeness and soundness: $(\mathcal{P}, V)$ achieves an (α, β) separation, where α and β are constants and $\alpha < \beta$.

zero-knowledge: It is easy to simulate the distribution on any 2 bits of P^*.

The zero-knowledge condition can be strengthened to work for any constant k, but the values of α and β will grow closer together as a function of k. Note that if V only probes 1 bit of the proof, then one can generate an optimal proof in random polynomial time. Thus, our result is essentially the best possible.

For our result on the hardness of computing probability intervals, we modify the construction of zero-knowledge transparent proofs. This construction, based on the construction of [5] makes heavy use of *randomizing tableaux* [12].

1.5 Intuition

In order to provide some intuition about the role of zero knowledge PCP's we provide a proof of Proposition 1 inspired by Goldreich, Micali and Wigderson's [11] (plain old) zero-knowledge protocol for proving 3-colorability of a graph. Their idea for proving colorability was for the prover to randomly permute the colors and then the verifier gets to check for a single edge whether it is properly colored. All the verifier sees is a random pair of distinct colors. This inspires the following:

To show the hardness we reduce graph 3-colorability to consistency checking. Given a graph $G = (V, E)$ define a variable X_{vc} for all $v \in V$ and $c \in \{1, 2, 3\}$ (the interpretation is that X_{vc} is true if node v is colored c). For every edge $(u, v) \in E$ and every $c \neq b \in \{1, 2, 3\}$ create the constraint

$$\Pr[X_{uc} \wedge X_{vb}] = \frac{1}{6}$$

Claim 6 The above set of constraints is consistent iff G is 3-colorable.

The proof of the claim is based on the completeness and soundness of the protocol of [11].

Remark: We note that there exist simpler proofs for weaker versions of our consistency checking result: For example, one may consider one-sided PCP's in which the verifier checks only three bits and accepts solely based on the bits queried and their values. The constraint set that assigns probability 0 to all rejecting views is consistent if a correct proof exists and ϵ-inconsistent if no such proof exists. This technique applies to systems where the constraints are on three boolean variables or two three-valued variables, but breaks when the constraints are only on two boolean variables. Indeed, historically the non-approximate versions of these theorems were quite easy for all but the two boolean variable case. For this last case, the original proof was quite complicated and used duality results from linear programming; our techniques later provided a much simpler proof. Furthermore, the zero-knowledge technique also applies when all pairwise events are assigned probabilities.

Outline of the paper

The rest of the paper is organized as follows. In Section 2 we show how to base consistency checking hardness results on zero-knowledge PCP's. In Section 3 we review randomizing tableaux. In Section 4 we give a method for reducing the number of bits queried to 2. In Section 5 we prove the NP-hardness of computing implied correlations.

2 Zero-knowledge PCP's and consistency checking

We establish a connection between zero-knowledge PCP's for NP and the NP-hardness of consistency checking. A PCP P can be viewed as a sequence of bits $P[1], \ldots, P[m]$, in which a query sequence $q = (q_1, \ldots, q_k)$ is answered with $a = (P[q_1], \ldots, P[q_k])$. We abbreviate this by $P[q] = a$. A "probabilistic PCP" $\mathcal{P}$ is a distribution on PCP's, and conceptually $(q_1, \ldots, q_k)$ is answered by $P[q_1], \ldots, P[q_k]$, where P is chosen according to $\mathcal{P}$. We write $\mathcal{P}[q] = a$ to denote the event that $P[q] = a$ when P is chosen according to $\mathcal{P}$. We assume for simplicity that all PCP's sampled according to $\mathcal{P}$ will be of the same length. Given a set of possible query sequences, and a probabilistic PCP, we define the following associated set of constraints.

Definition 4 Let $\mathcal{P}$ be a probabilistic PCP. We define $C^k_{\mathcal{P}}$ to be the set of constraints of the form

$$\Pr[x_{q_1} = a_1, \ldots, x_{q_k} = a_k] = p_{q,a},$$

where $q_i \in [1, |P|], a_i \in \{0, 1\}$ and $p_{q,a}$ is the probability that $P[q_i] = a_i$ for $1 \leq i \leq k$, when P is chosen according to $\mathcal{P}$.

Note that $C^k_{\mathcal{P}}$ is a k-simple set of constraints. If V makes only k queries to $\mathcal{P}$, then we observe that $C^k_{\mathcal{P}}$ contains all of the information we need to compute the probability that V accepts $\mathcal{P}$. We can simulate V checking $\mathcal{P}$ by running V until it comes up with questions $q = (q_1, \ldots, q_k)$, and then sampling $a = (a_1, \ldots, a_k)$ with probability $p_{q,a}$. By our definition, this is the exact view that V obtains in querying $\mathcal{P}$. We will therefore speak of the probability that (C, V) accepts, which is well defined even when C is inconsistent. In general, one can efficiently estimate the probability that (C, V) accepts by running many simulations. For the proofs, constraints and verifiers we construct, one can determine the acceptance probabilities exactly.

In general, given a description of $\mathcal{P}$ it may be very difficult to compute $C^k_{\mathcal{P}}$. However, it is very easy to compute $C^k_{\mathcal{P}}$ for special cases of zero-knowledge PCP's. Suppose that the simulator can perfectly simulate the view obtained by a verifier who may make an arbitrary set of k queries. Then one can compute $C^k_{\mathcal{P}}$ by considering the equivalent distribution produced by the simulator. Using this observation, we construct examples where determining the existence of $\mathcal{P}$ is NP-hard, but for which computing $C^k_{\mathcal{P}}$ (assuming $\mathcal{P}$ exists) is easy.

By definition, $\mathcal{P}$ satisfies $C^k_{\mathcal{P}}$. Now suppose that some PCP $\mathcal{P}'$ and constraint set C, $\mathcal{P}'$ $(1 - \epsilon)$-satisfies C. Lemma 7 says that if a verifier V only queries k bits, then (C, V) and $(\mathcal{P}', V)$ will accept with similar probabilities.

Lemma 7 Let V, C and $\mathcal{P}'$ be as above. Then

$$|\Pr[(C, V) \text{ accepts}] - \Pr[\mathcal{P}', V]| \leq 2^k \epsilon$$

Proof: Let $p_V(q)$ denote the probability that V makes query q and let $V(q, a)$ be the probability that V accepts given that it made query q and received answer a. Then $|\Pr((C, V) \text{ accepts}) - \Pr((\mathcal{P}', V) \text{ accepts})|$ can be bounded above by

$$\sum_{q,a} p_V(q) V(q, a) \left|\Pr[\mathcal{P}'[q] = a] - \Pr[C[q] = a]\right|$$

$$\begin{aligned}
&\leq \sum_{q,a} p_V(q)V(q,a)\epsilon \\
&\leq \sum_{q} p_V(q)\epsilon 2^k \\
&= \epsilon 2^k
\end{aligned}$$

□

Let $L \in NP$, and let $(\{\mathcal{P}_x\}, V)$ denote a family of probabilistic PCP's for $x \in L$ that achieves an (α, β) separation, and in which V makes k non-adaptive queries. Lemma 8 shows how to test membership in L if one can construct a certain family of k-simple constraints and then test approximate consistency on these constraints.

Lemma 8 Let $(\{\mathcal{P}_x\}, V)$ be as above and let C_x be a set of k-simple constraints such that

- $(C_x, V(x))$ accepts with probability at least β.
- If $x \in L$, then $C_x = C^k_{\mathcal{P}_x}$.

Then either $x \in L$ and C_x is consistent, or $x \notin L$ and C_x is ϵ-inconsistent for any $\epsilon < 2^{-k}(\beta - \alpha)$.

Proof: By definition, C_x is consistent for all $x \in L$. It suffices to show that if C_x is $(1-\epsilon)$-consistent for $\epsilon < 2^{-k}(\beta - \alpha)$, then $x \in L$. Let $\mathcal{P}'$ be the probabilistic PCP that $(1-\epsilon)$-satisfies C_x. By Lemma 7,

$$\begin{aligned}
|\Pr[(C,V) \text{ accepts}] - \Pr[(\mathcal{P}',V) \text{ accepts}]| &\leq 2^k\epsilon \\
&< \beta - \alpha.
\end{aligned}$$

Hence,

$$\begin{aligned}
\Pr[(\mathcal{P}',V) \text{ accepts}] &> \Pr[(C,V)\text{accepts}] - (\beta - \alpha) \\
&> \alpha.
\end{aligned}$$

Hence, there is a prover that makes the verifier accept with probability greater than α, so $x \in L$.

□

Thus, if one can efficiently generate C_x from x, and distinguish between the case where C_x is consistent and C_x is ϵ-inconsistent for some $\epsilon < 2^{-k}(\beta - \alpha)$, then one can determine membership in L. For any $L \in NP$, we exhibit a probabilistic PCP family $(\{\mathcal{P}_x\}, V)$ such that C_x is easy to compute, where $k = 2$. This implies Theorem 1.

Finally, for the weighted case, we note that a zero-knowledge PCP also assigns a natural weighting ω to the events in the constraint set. For each event, simply assign the probability that this is the view of the verifier (i.e. that the verifier looks at these bits and sees the specified values). In this case, if two measures are close according to ω, the corresponding probabilities of acceptance will also be close. Thus, the construction that proves Theorem 1 will also prove Theorem 2. Details are omitted from this extended abstract.

3 Randomizing Tableaux

Randomizing tableaux were introduced in [12] and more recently were used in the creation of "zero-knowledge" PCP's in [5]. They allow one to make proofs that remain zero-knowledge as long as sufficiently few bits of the proof have been read. In this abstract, we describe the properties of randomizing tableaux that we will use. A detailed discussion of randomizing tableaux may be found in [13, 5].

Let k be a fixed constant (for our applications, k will be 2 or 3). Given variables $X = x_1, \ldots, x_n$ and 3SAT expression $S = C_1 \wedge \cdots \wedge C_m$, the randomizing tableau construction first expands each bit x_i into a random exclusive-or of bits by

$$x_i = x_{i,1} \oplus x_{i,2} \oplus \ldots \oplus x_{i,k}$$

The randomizing tableaux construction produces for each clause C a tableau $(\{g_{i,j}^C\}, \{r_{i,j}^C\}, \{c_{i,j}^C\})$, where, following Barrington [2], $g_{i,j}^C, r_{i,j}^C, c_{i,j}^C \in S_5$, where S_5 denotes the group of permutations on 5 elements. The properties we use are the following:

1. The tableau may be checked by performing a random test, each of which consists of a constant number of queries to g^C, r^C and c^C and $x_{i,j}$. If X does not satisfy S, a random test will catch this fact with probability $\Omega(1)$.
2. Each test reveals information about at most one value of $x_{i,j}$.
3. If one performs a set of less than k checks, one can easily simulate the distribution on the observed elements of the tableau.

To check whether X satisfies S, the verifier picks a random clause and then picks a random test for C's tableau. This involves looking at $O(1)$ (28, to be precise) bits of the tableau. We can augment the tableau to reduce this number to 3 as follows.[1] For each test, write down a fan-in 2 circuit that computes the result of this test. Then write down the output of each gate of the circuit. To test the augmented tableau, the verifier picks a random test, and then check the consistency of the inputs and outputs of a random gate in the circuit (making sure that the final gate outputs 1). Since the circuit for computing a test is of constant size, this degrades the probability of catching an incorrect tableau by only a constant factor. We make two observations that we will need for our proof:

1. The final proof contains the values of $x_{i,k}$ (since bits are only added)
2. Any auxiliary bit only reveals information about one test. Hence, any k bits of the augmented proof only reveals information about k tests in the original proof.

4 Zero-Knowledge with Two-Bit Queries

We review the result that will appear in the journal version of [5], and which follows from the previous discussion. Let S be a 3SAT expression, $C_1 \wedge \cdots \wedge C_m$ over variables $x_1, \ldots, x_n$.

[1] This construction will appear in the journal version of [5]. We include it because we need to use details of the construction.

Then the techniques of Dwork et al., allow us to efficiently construct a 3SAT expression $S' = C'_1 \wedge \cdots \wedge C'_{m'}$ over variables $y_1, \ldots, y_{n'}$, where $m', n' = (mnk)^{O(1)}$, with the following properties:

("completeness") If S is satisfiable, then S' is satisfiable.

("soundness") If S is not satisfiable, then one cannot simultaneously satisfy more than $(1-\delta)m'$ of the clauses of S', where δ depends only on k.

("zero-knowledge") Given a satisfying assignment for S, one can randomly generate a satisfying assignment for S' such that for any $i_1, \ldots, i_k$, the induced distribution on the set of variables mentioned in clauses $C_{i_1}, \ldots, C_{i_k}$ is easy to compute.

The completeness, soundness and zero-knowledge properties in the [5] protocol follow directly from the corresponding properties of S'.

Given a satisfying assignment $x = x_1, \ldots, x_n$ for S, one can sample from a zero-knowledge probabilistic PCP for S' as follows. We denote clause C'_i by $C'_i = (t_{i1} y_{\ell_{i1}} \vee t_{i2} y_{\ell_{i2}} \vee t_{i3} y_{\ell_{i3}})$, where t_{ij} denotes the presence or lack of negation.

1. Generate a satisfying assignment $y = y_1, \ldots, y_{n'}$.

2. Let $z_i \in \{1,2,3\}$ denote the smallest j such that y satisfies $t_{ij} x_{\ell_{ij}}$. The resulting PCP is given by $y_1, \ldots, y_{n'}, z_1, \ldots, z_{m'}$.

To verify S, V chooses $i \in_R [1, m'], j \in_R \{1,2,3\}$ and reads z_i and $y_{\ell_{ij}}$. V rejects if $z_i = j$ and $t_{ij} y_{\ell_{ij}}$ is false, and otherwise accepts. By a straightforward case analysis, V accepts if S is satisfiable, and rejects with probability $\delta/3$ is S is not satisfiable.

4.1 Implementing the 3-bit protocol with 2 bit queries.

In the protocol of [5], V looks at a trit and a bit of the proof. This is optimal if one requires perfect completeness, but we can modify it so that the verifier only needs to look at two bits of the proof. This new proof achieves an (α, β) separation, where $\beta - \alpha = \Omega(\delta)$.

Let $S', y_1, \ldots, y_{n'}, z_1, \ldots, z_{m'}$ be as above. For $1 \leq i \leq m'$ and $1 \leq j \leq 3$, define $z_{ij} = 1$ if $z_i = j$ and $z_{ij} = 0$ otherwise. The new PCP is given by $y_1, \ldots, y_{n'}, z_{11}, \ldots, z_{m'3}$.

To verify the proof, V first chooses $i \in_R [1, m']$ and $j \in_R \{1,2,3\}$, then makes one of the following two tests:

1. With probability $\frac{2}{3}$, V reads z_{ij} and $y_{\ell_{ij}}$. V rejects if $z_{ij} = 1$ and $t_{ij} y_{\ell_{ij}}$ is false and accepts otherwise.

2. With probability $\frac{1}{3}$, V chooses $k \in_R \{1,2,3\} - \{j\}$, and reads z_{ij} and z_{ik}. V rejects if $z_{ij} = z_{ik}$ and accepts otherwise.

4.2 Analysis of the proof system

We show that the above protocol achieves (α, β) separation, for suitable α and β, and that it achieves zero-knowledge for a bounded number of probes.

Lemma 9 If S is satisfiable, and the proof is correctly constructed, then V will accept with probability $8/9$. If S is not satisfiable, then V will accept with probability at most $8/9 - 2\delta/9$.

Proof:

Let c_i denote the number of distinct j such that $z_{ij} = 1$. The probability that V will reject on performing Test 2 is 1 if $c_i = 0$, $\frac{1}{3}$ if $c_i = 1$ or $c_i = 2$ and 1 if $c_i = 3$. If $y_1, \ldots, y_{n'}$ doesn't satisfy clause C_i' then the probability that V will reject on performing Test 1 is $c_i/3$.

If S is satisfiable, and the proof has been properly constructed, then by the definition of z_{ij}, V will never reject based on Test 1. Furthermore, for every i, $c_i = 1$ so if V runs Test 2, he will reject with probability $\frac{1}{3}$. Since Test 2 is run with probability $\frac{1}{3}$, V will reject with probability 1/9, and thus accept with probability 8/9.

If S is not satisfiable, then let I denote the set of indices such that $y_1, \ldots, y_{n'}$, does not satisfy clause C_i' of S'. By the construction of S', $|I|/m' \geq \delta$. For $i \in I$, we perform the following case analysis on c_i for the probability that V will reject:

$$\begin{array}{lcccc}
c_i = 0 & : & \frac{2}{3} \cdot 0 + \frac{1}{3} \cdot 1 & = & \frac{1}{3} \\
c_i = 1 & : & \frac{2}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{3} & = & \frac{1}{3} \\
c_i = 2 & : & \frac{2}{3} \cdot \frac{2}{3} + \frac{1}{3} \cdot \frac{1}{3} & = & \frac{5}{9} \\
c_i = 3 & : & \frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 1 & = & 1
\end{array}$$

Hence, V will reject with probability at least $\frac{1}{3}$ whenever $i \in I$. When $i \notin I$, V will run Test 2 with probability $\frac{1}{3}$ and thus reject with probability at least 1/9. The probability that V will reject can therefore be bounded below by

$$\frac{1}{3}\left(\frac{|I|}{m'}\right) + \frac{1}{9}\left(1 - \frac{|I|}{m'}\right) \geq \frac{1}{9} + \frac{2}{9}\delta.$$

Lemma 10 It is easy to compute the distribution on any k bits of the probabilistic PCP.

Since each bit y_i is the same as those that appear in the construction of Dwork et al. and z_{ij} depends solely on z_i in that construction, the distribution of any k bits of the new construction can be computed trivially from the distribution of k values of the Dwork et al. construction (some of these values may be trits). Hence, the lemma follows from the simulatability of the Dwork et al. construction. □

5 On the difficulty of computing probability intervals and implied correlations.

We now outline how to alter the proof for the hardness of finding statistical inconsistencies to one for the hardness of computing probability intervals and implied correlations, thus proving Theorem 3 and Theorem 4. Given a 3-SAT expression $S = C_1, \ldots, C_m$, we create a new variable y_α, and consider the new expression $S^* = C_1^*, \ldots, C_m^*$, where if $C_i = (y_{i,1} \vee y_{i,2} \vee y_{i,3})$ then $C_i^* = (y_\alpha \vee y_{i,1} \vee y_{i,2} \vee y_{i,3})$. If S is satisfiable, then there are no restrictions on y_α. If S is not satisfiable, then S^* is still satisfiable, but all satisfying solutions have $y_\alpha = 1$.

Now, when we create the randomizing tableaux proof, with parameter $k = 2$, we break y_α into an exclusive-or, $y_\alpha = y_{\alpha,1} \oplus y_{\alpha,2}$. Consider the set $\mathcal{C}$ of constraints corresponding to all of the legal tests and the additional constraints

$$\Pr[y_{\alpha,1} = 0] = \Pr[y_{\alpha,1} = 1] = \frac{1}{2}$$

(in fact, these constraints are implied by the other set of constraints, and are included to clarify the exposition).

By a straightforward argument, we have

$$\begin{aligned} \Pr[y_\alpha = 0] &= \frac{Corr(y_{\alpha,1}, y_{\alpha,2}) + 1}{2} \text{ and} \\ \Pr[y_\alpha = 0] &= 2 \cdot \Pr[y_{\alpha,1} \wedge y_{\alpha,2}]. \end{aligned}$$

Now, if S is satisfiable, then one can choose $y_\alpha = 0$ with whatever probability one wishes, pick the other variables so as to satisfy S', and then go through the randomized proof construction. In this case $Corr(y_{\alpha,1}, y_{\alpha,2})$ can be made to be made to take on any value in $[-1, 1]$ and $\Pr[y_{\alpha,1} \wedge y_{\alpha,2}]$ can be made anything from 0 to $\frac{1}{2}$. However, if S is not satisfiable, then for any distribution it must follow that $y_\alpha = 1$ with probability 1. Otherwise, one of the constraints of the randomizing tableau will be violated, which will ultimately lead to a nonzero probability of a "bad event" that should in fact happen with probability 0. This implies that $Corr(y_{\alpha,1}, y_{\alpha,2}) = -1$ and $\Pr[y_{\alpha,1} \wedge y_{\alpha,2}] = 0$, since $(0, 1)$ and $(1, 0)$ are the only admissible values for $(y_{\alpha,1}, y_{\alpha,2})$. $\square$

6 Open Questions

A natural open question to ask is whether one can find a more elementary argument for Theorem 1. The constant obtained by the current technique is currently quite small, and would most likely be improved by an argument that didn't use current PCP results. Also, suppose one only considers the maximum entropy distribution that satisfies a given set of constraints. It would be important to understand the complexity of reasoning about this distribution. For example, can one compute (or approximately compute) the probability of $k + 1$-wise events of the maximum entropy distribution satisfying a given set of k-simple constraints?

7 Acknowledgments

We thank Nir Friedman, Daphne Koller for very useful discussions. We thank Ran Canetti and anonymous referees for reading the paper and providing helpful advise.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, *Proof verification and intractability of approximation problems*, Proc. 33^{rd} IEEE Symp. on Foundation of Computer Science, 1992.

[2] D. A. Barrington, *Bounded-width polynomial-sized branching programs recognize exactly those languages in NC^1*, J. of Computer and Systems Sciences, vol 38, 1988, pp. 150-164.

[3] M. Bellare, S. Goldwasser, C. Lund, A. Russell, *Efficient probabilistic checkable proofs and applications to approximation, Proc. 25^{th} STOC, 1993, pp. 294-304.*

[4] E. Charniak, *Bayesian networks without tears*, AI Magazine, 1991, pp. 50–63.

[5] C. Dwork, U. Feige, J. Kilian, M. Naor, S. Safra, *Low communication 2-prover zero-knowledge proofs for NP* Advances in Cryptology – Proceedings of Crypto '92, Lecture Notes in Computer Science 740, Springer Verlag, 1993, pp. 215-227.

[6] G. F. Cooper, *The computational complexity of probabilistic inference using Bayesian belief networks*, Artificial Intelligence **42**, 1990, pp. 393–405.

[7] Dagum, P. and Luby, M. *Approximating probabilistic inference in Bayesian belief networks is NP-hard*, Artificial Intelligence (March 1993) vol.60, no.1, pp. 141–53.

[8] R. Fagin, J. Halpern and N. Megiddo, *A Logic for Reasoning About Probabilities*, Information and Computation **87**, 1990, pp. 78–128.

[9] U. Feige, S. Goldwasser, L. Lovasz, M. Safra, M. Szegedy, *Approximating Clique is Almost NP-Complete* Proc. 32^{nd} IEEE Symp. on Foundation of Computer Science, 1991, pp. 2-12.

[10] U. Feige and J. Kilian, *Two Prover Protocols – Low Error at Affordable Rates*, Proceedings 26^{th} Annual Symposium on the Theory of Computation, 1994.

[11] O. Goldreich, S. Micali, A. Wigderson, *Proofs that Yield Nothing but Their Validity and a Methodology of Cryptographic Protocol Design*, J. of the ACM 38, 1991, pp. 691–729.

[12] J. Kilian. *Founding Cryptography on Oblivious Transfer*, Proceedings 20th Annual Symposium on the Theory of Computation, 1988.

[13] J. Kilian, **Use of Randomness in Algorithms and Protocols**, MIT Press, 1990.

[14] D. Koller, N. Megiddo, "Constructing small sample spaces satisfying given constraints", *Proc. of 25^{th} STOC, 1993, pp. 268-277.*

[15] N. Nilsson, *Probabilistic Logic*, Artificial Intelligence **28**, 1986, pp. 71-87.

[16] E. Neopolitan, **Probabilistic reasoning in expert systems**, Wiley, New York, 1990.

[17] J. Pearl, **Probabilistic reasoning in intelligent systems: networks of plausible inference**, Morgan Kaufman, San Mateo, 1988.

[18] D. Roth, *On the Hardness of Approximate Reasoning*, Artificial Intelligence Journal, Volume 82, 1-2, pp 273–302, April 1996.

[19] L. G. Valiant, *The complexity of computing the permanent*, Theoretical Computer Science, 8:189–201, 1979.

[20] L. G. Valiant, *The complexity of enumeration and reliability problems*, SIAM Journal of Computing, 8:410–421, 1979.