A Data Prism: Semi-Verified Learning in the Small-α Regime

Michela Meister MMEISTER@STANFORD.EDU and **Gregory Valiant** VALIANT@STANFORD.EDU
Stanford University

Editors: Sebastien Bubeck, Vianney Perchet and Philippe Rigollet

Abstract

We consider a simple model of unreliable or crowdsourced data where there is an underlying set of n binary variables, each "evaluator" contributes a (possibly unreliable or adversarial) estimate of the values of some subset of r of the variables, and the learner is given the true value of a *constant* number of variables. We show that, provided an α-fraction of the evaluators are "good" (either correct, or with independent noise rate $p<1/2$), then the true values of a $(1-\epsilon)$ fraction of the n underlying variables can be deduced as long as $r>\log_{2-2p}(1/\alpha)$. For example, if the fraction of "good" evaluators is larger than $1/16$ and there is no noise in their responses, then accurate recovery is possible provided each worker evaluates a random set of 4 items. This result is optimal in that if $r\leq\log_{2-2p}(1/\alpha)$ the large dataset can contain no information. This setting can be viewed as an instance of the *semi-verified* learning model introduced in (Charikar et al., 2017), which explores the tradeoff between the number of items evaluated by each worker and the fraction of "good" evaluators. In the standard adversarial setting, our algorithm requires $\tilde{O}\big(n^{\log_{2-2p}(1/\alpha)}\big)$ evaluators. However, the algorithm runs in near linear time, $\tilde{O}_{r,\epsilon}(n)$, and hence would require only a near-linear number of evaluations in the weaker model in which the adversary's responses to each r-tuple of items are independent of the set of evaluations collected. These settings and results can also be viewed as examining a general class of semi-adversarial CSPs with a planted assignment.

This extreme parameter regime, where the fraction of reliable data is small (inverse exponential in the amount of data provided by each source), is relevant to a number of practical settings. For example, the setting where you collect a dataset on customer preferences, with each customer specifying preferences for a small (constant) number of items, and the goal is to ascertain the preferences of a specific demographic of interest. Our results show that this large dataset (which lacks demographic information) can be leveraged together with the preferences of the demographic of interest for a *constant* (polynomial in $1/\alpha$ but independent of n), number of randomly selected items, to recover an accurate estimate of the entire set of preferences, even if the fraction of the original dataset contributed by the demographic of interest is inverse exponential in the number of preferences supplied by each customer. In this sense, our results can be viewed as a "data prism" allowing one to extract the behavior of specific cohorts from a large, mixed, dataset.

Keywords: Semi-Verified Learning, Robust Statistics, Cohort Analysis, Crowdsourcing

1. Introduction

Imagine that you collect a large dataset of market research. Specifically, the dataset consists of customer evaluations of products. While the total set of products is large, of size n, each customer is only asked to evaluate a small (perhaps randomly selected) subset of $r=2$,3, etc. of those products. Now suppose you wish to identify the preferences of some special demographic of customers—perhaps the customers who are full-time students. Let α denote a lower bound on the fraction of the surveyed customers that were full-time students, but assume that we do not have this demographic information in our dataset–all we have is the set of evaluations of each customer. How can we leverage this dataset to learn anything about the student-demographic?

If $\alpha \ll 1/2$, this problem seems hopeless because the amount of data contributed by non-students might swamp the portion of the dataset contributed by students (the demographic of interest). Nevertheless, the main result of this paper shows that one could hire some students to evaluate a *constant*, k, number of (random) products in the set of size n, and then leverage that constant amount of information together with the large dataset to return accurate evaluations of the student-demographic preferences on all n items. This claim will hold provided the number of items evaluated by each of the customers in the dataset, $r > \log_2(1/\alpha)$. The guarantees of the algorithm will ensure that, with high probability, at most an ϵ-fraction of the returned evaluations are incorrect (where k—the number of products evaluated by the hired students, is polynomial in $1/\epsilon$ and $1/\alpha$, but is *independent* of the total number of items, n). In particular, this strong success guarantee holds irrespective of the behavior of the non-student demographics in the original dataset–in particular, they could even be adversarial, provided by a single malicious entity who is trying to disguise the feedback provided by the student-demographic.

The above setting, where one has a large dataset reflecting a number of demographics, and wishes to leverage the large dataset in conjunction with a very small set of "verified" datapoints from one demographic of interest, seems relevant beyond the market research domain. Indeed, there are many biological or health-related datasets where the "demographic of interest" might be a trait that is expensive to evaluate such as genetic factors.

While the parameter regime explored in this work are inspired by these real-world settings, there are also close connections to several classes of constraint-satisfaction-problems (CSPs). The connections we present between our crowdsourcing setting and CSPs may offer a new perspective on problems in these areas. Indeed, the purely combinatorial approach of our algorithm is in contrast to the geometric or spectral methods employed by previous approaches to robust statistics and crowdsourcing, and seems essential to obtaining a super-polynomial tradeoff between the fraction of "good" data and the number of evaluations requested per evaluator.

1.1. Formal Model

We formally model this problem as an instance of the *semi-verified learning* model proposed by Charikar, Steinhardt, and Valiant (Charikar et al., 2017). Suppose there is a set of n Boolean variables, $V = \{v_1,...,v_n\}$, and m "workers" who each provide an evaluation of the values of a randomly selected subset of r of the variables. Suppose that an α-fraction of the workers are "reliable" and submit evaluations with the property that each of their r reported values is incorrect independently with probability $\leq p$. We make no assumptions on the evaluations submitted by the $(1-\alpha)m$ unreliable workers—these evaluations could be biased, arbitrary, or even adversarially chosen with the goal of confounding the learning algorithm. In addition to this large dataset, we also receive $k \ll n,m$ "verified" data points that consist of the values of k randomly selected variables. The goal of the learner will be to return assignments to the n variables, such that with probability at least $1-\delta$, at most ϵn of these returned assignments differ from their true values.

In the first model we consider, we have a dataset containing evaluations for each of the $\binom{n}{1+\lfloor \log_{2-2p}(1/\alpha)\rfloor}$ sets of $1 + \lfloor \log_{2-2p}(1/\alpha)\rfloor$ items/variables. This corresponds to soliciting responses from at least $m = \tilde{O}\left(n^{1+\lfloor \log_{2-2p}(1/\alpha)\rfloor}\right)$ workers. We assume that an adversary selects the responses given by the $(1-\alpha)$-fraction of "bad" workers, and that we have query access to the dataset, so that, for any set t of $\leq r$ products, we can query the evaluations from all workers who reviewed the set of products, t, in constant time. In this setting, we show that, provided $r \geq 1 + \lfloor \log_{2-2p}(1/\alpha)\rfloor$, given the $k = O_{\alpha,\epsilon,\delta}(1)$ "verified" evaluations [e.g. from the hired students], we can recover the correct labels [the true preferences of the student demographic] on a $(1-\epsilon)n$ fraction of the items/variables after making $\tilde{O}_{\alpha,\epsilon,\delta}(n)$ queries to the dataset and in linear time, $\tilde{O}_{\alpha,\epsilon,\delta}(n)$, with probability at least $1-\delta$ over the randomness in the algorithm and in the noise in the "good" workers' responses.

The requirement that we have evaluations for all subsets of $1+\lfloor \log_{2-2p}(1/\alpha)\rfloor$ variables may be unrealistic in practice. However, as our algorithm only queries a linear number of such subsets (and does so non-adaptively), the results continue to hold in the setting where only $\tilde{O}_{r,\epsilon,\delta}(n)$ workers are required, provided that the responses of the adversaries do not adaptively depend on the choice of tuples evaluated. This is equivalent to the adversary first determining the responses to all $\binom{n}{r}$ possible sets of r items, before we select which $\tilde{O}_{r,\epsilon,\delta}(n)$ tuples to have evaluated by the workers. This model, while weaker than the fully adversarial setting, corresponds to the "data prism" setting where the "adversarial" data may be arbitrarily biased, but does not correspond to an adaptive adversary.

1.2. Summary of Results and Connections to Random CSPs

Our main result is the following:

Theorem 1 *Fix a failure probability $\delta > 0$ and accuracy parameter $\epsilon > 0$. Consider a set of n items that each have a Boolean value, and m reviewers who each evaluate a uniformly random subset of r out of the n items. Suppose that αm of the reviewers are "good" in that each of their r reviews is correct (independently) with probability at least $1-p \geq 1/2$. Given sufficiently many reviewers, accurate reviews of at least $(1-\epsilon)n$ items can be inferred given the true values of a constant (independent of n) sized random subset of the variables, provided the fraction of good reviewers satisfies $\alpha > \frac{1}{(2-2p)^r}$.*

Specifically, given the values of a random, constant-sized subset of the items of size $k = \tilde{O}\left(\frac{1}{\epsilon}\cdot(1/\alpha)^{\log_{2-2p}4}\log(1/\delta)\right)$, with probability at least $1-\delta$ one can recover accurate evaluations of at least $(1-\epsilon)n$ of the items, provided $\alpha > \frac{1}{(2-2p)^r}$ and the number of reviewers $m = \tilde{\Theta}_{\alpha,\delta,\epsilon}\left(n^{1+\lfloor \log_{2-2p}(1/\alpha)\rfloor}\right)$.

Additionally, the algorithm runs in time linear in the number of items, n, given the ability to query the dataset for reviewers who have evaluated a given set of items in constant time. Specifically, the runtime of the algorithm is $O_{\delta,\epsilon,r}(n)$, where the hidden constant hides a polynomial dependence on $1/\alpha$,$1/\epsilon$ and $\log(1/\delta)$.

As a consequence of the structure of the algorithm to which the above theorem applies, the following theorem captures the sense in which a linear number of reviewers are necessary if 1) reviewers can be assigned items to review (rather than reviewing uniformly random subsets) and 2) the strategy of the adversary is independent of the set of reviews requested.

Theorem 2 *Given the ability to request evaluations of sets of $r \geq 1+\lfloor \log_{2-2p}(1/\alpha)\rfloor$ items from each worker, and assuming that the responses of the "bad" fraction of workers on a given set of items is independent of the set of other queries asked, accurate estimates of a $(1-\epsilon)n$ fraction of items can be obtained with $m = \tilde{O}_{\alpha,\epsilon,\delta}(n)$ workers and $k = poly(1/\alpha, 1/\epsilon, \log(1/\delta))$ verified evaluations.*

In the case where the m workers can be queried after receiving the k verified evaluations, the hidden dependence of m on α,ϵ,δ is polynomial. If the m workers are queried before receiving the k verified evaluations, then this dependence is exponential but still independent of n, the number of items. This exponential dependency corresponds to collecting queries that would correspond to each of the 2^k possible outcomes of the k verified evaluations.

The following straightforward observation demonstrates that the above theorems are optimal in the relationship between the fraction of good reviewers, α, and the number of items reviewed by each individual, r, and the error rate of each good reviewer, p:

Observation 1 *If each good reviewer incorrectly reviews each item independently with probability p, and the fraction of good reviewers satisfies $\alpha = \frac{1}{(2-2p)^r}$ where r denotes the number of items evaluated by each reviewer, then the remaining $(1-\alpha)$ fraction of reviewers can behave such that for every set of r items, for a randomly selected reviewer, the distribution of reviews for those items will be uniform over the 2^r possible review vectors. Hence the dataset contains no useful information.*

One reason why Theorem 1 is surprising is that this inverse exponential dependence between the number of reviews per reviewers, r, and the fraction of "good" reviewers, can not be attained via the usual approach of low-rank matrix approximation that is often applied to this problem of recommendation systems (e.g. (Candes and Plan, 2010; Keshavan et al., 2010)). To see why these approaches cannot be applied, note that for any matrix in which all rows have at most r entries, there is a rank r matrix that exactly agrees with all entries. Intuitively, each of these r factors is capable of representing a different subset of the reviewers. Still, at best this would result in an algorithm that is capable of capturing r different groups of reviewers; in other words, it seems extremely unlikely that such approaches could yield positive results in the setting where the fraction of "good" reviewers was less than $1/r$, in contrast to our results that allow this fraction to be $1/exp(r)$.

The setting of Theorem 1 can be easily mapped into the language of a constraint satisfaction problem. Given the evaluations of the reviewers, we build a constraint satisfaction problem by associating a Boolean variable to each of the n items, and for every set of r variables, we define the set of allowable assignments to those variables to include any of the 2^r review vectors that constitutes more than a $1/2^r$ fraction of the review vectors for the associated items. (In other words, if at most a $1/2^r$ fraction of the reviewers who evaluated a given set of r items submitted a vector of reviews $\sigma = (\sigma_1,...,\sigma_r)$, then σ is not an allowable assignment for those variables.) The requirement that $\alpha > \frac{1}{(2-2p)^r}$ guarantees that, for every set of r items, irrespective of the behaviors of the $(1-\alpha)$ fraction of bad reviewers, for a randomly selected reviewer, the probability that the r reviews are all correct is strictly larger than $1/2^r$. Additionally, our requirement on the number of reviewers, m, ensures that with probability $> 1-\delta$, (by elementary concentration bounds) for every set of r items, there are sufficiently many reviewers assigned to that set of r items, so as to ensure that the number of accurate ratings (provided by the good reviewers) exceeds a $1/2^r$ fraction of the overall reviews for that set of r items. Hence, with high probability, we obtain a constraint satisfaction problem such that for every set of r variables 1) the correct assignment is in the set of allowable assignments, and 2) at least one of the 2^r possible assignments is disallowed.

Given this mapping from the review/evaluation setting to constraint satisfaction problems, Theorem 1 will follow immediately from the following result concerning a class of adversarial constraint satisfaction problems:

Theorem 3 *Consider a set of n Boolean variables, and a planted assignment $\sigma \in \{0,1\}^n$. Suppose that for each subset of r variables, $t = \{v_1,...,v_r\}$, there is a subset $C_t \subset \{0,1\}^r$ of assignments such that $|C_t| \leq 2^r - 1$ and the planted assignment σ (restricted to the variables in t) is in set C_t. Given the ability to query the planted assignment values for a constant number of variables chosen uniformly at random, the planted assignment can be recovered with up to ϵn errors, for any constant $\epsilon > 0$.*

Specifically, for any $\epsilon,\delta > 0$, after querying the values of

$$k = \tilde{O}\left(\frac{1}{\epsilon} \cdot 2^{2r} \log(1/\delta)\right)$$

variables, with probability at least $1-\delta$ we can output an assignment $\sigma' \in \{0,1\}^n$ that differs from the planted assignment, σ, in at most ϵn values. Additionally, the algorithm will run in time $O_{r,\epsilon,\delta}(n)$.

There is a simple VC-dimension argument together with a sphere-packing result of Haussler (Haussler, 1995) that yields a tighter information theoretic recovery result, yielding an analog of the above theorem with polynomial dependence on r.[1] Specifically, the number of verified assignments must be $k = O(\frac{1}{\epsilon}(r\log(1/\epsilon) + \log(1/\delta))$. This VC-dimension approach, however, seems to yield an algorithm with runtime at least n^r, as opposed to the linear time algorithms of Theorems 1 and 3. For practical settings, having a linear-time algorithm seems quite important; that said, exploring this problem from an information theoretic perspective is also worthwhile. One natural question is whether one can achieve a best-of-both-worlds: a near-linear time algorithm with a polynomial dependence on r. We discuss this problem more in Section 3.

Proposition 4 *As in Theorem 3, consider a set of n Boolean variables, and a planted assignment $\sigma \in \{0,1\}^n$. Suppose that for each subset of r variables, $t = \{v_1,\ldots,v_r\}$, there is a subset $C_t \subset \{0,1\}^r$ of assignments such that $|C_t| \le 2^r - 1$ and the planted assignment σ (restricted to the variables in t) is consistent with C_t. Given the ability to query the planted assignment values for $k = O\left(\frac{1}{\epsilon}(r\log(1/\epsilon) + \log(1/\delta))\right)$ random entries, with probability at least $1-\epsilon$ one can recover an assignment that disagrees with σ on at most ϵn values.*

Proof Let $S \subset \{0,1\}^n$ be the set of assignments that are consistent with all of the sets of partial assignments to the r-tuples specified by the sets C_t. The Vapnik-Chervonenkis (VC) dimension of the set S is at most r, since, by assumption, for every r-tuple of variables, $t = \{v_1,\ldots,v_r\}$, there are at most $|C_t| \le 2^r - 1$ possible assignments to those variables. As was shown by Haussler (Theorem 1 in (Haussler, 1995)), for any subset S of the Boolean hypercube with VC dimension at most r, for every $\epsilon > 0$ there exists a set $T \subset \{0,1\}^n$ of size at most $e(r+1)\left(\frac{2e}{\epsilon}\right)^r$ such that for every point $x \in S$, there exists a point $t_x \in T$ that agrees with x on at least $(1-\epsilon)n$ coordinates.

Let $T_{\epsilon/2}$ denote such a covering set corresponding to the set S, such that every $x \in S$ is distance at most $n\epsilon/2$ from an element of $T_{\epsilon/2}$. We can use our $k = O\left(\frac{1}{\epsilon}(r\log(1/\epsilon) + \log(1/\delta))\right)$ random coordinates of the vector $\sigma \in S$ to find, with probability at least $1-\delta$, a point in $T_{\epsilon/2}$ of distance at most $n\epsilon$ from σ by simply choosing the element of $T_{\epsilon/2}$ that agrees with the largest fraction of the k random samples. This follows from 1) leveraging a Chernoff bound to show that out of the k samples, at most a $(2/3)\epsilon$ fraction will disagree with the element of $T_{\epsilon/2}$ that has distance $\epsilon n/2$, and 2) a union bound over $|T_{\epsilon/2}|$ Chernoff bounds to argue that none of the elements of $T_{\epsilon/2}$ that have distance at least ϵn will disagree in fewer than a $(2/3)\epsilon$ fraction of indices. Together, this yields that the probability that the element of $T_{\epsilon/2}$ that agrees with the largest fraction of the k random samples has distance greater than ϵn from the true assignment, is at most $|T_{\epsilon/2}| exp(O(-k\epsilon)) = |T_{\epsilon/2}|(1/\epsilon)^{O(r)}\delta$, which is at most δ for a suitable choice of the constant in the "O" term of $k = O\left(\frac{1}{\epsilon}(r\log(1/\epsilon) + \log(1/\delta))\right)$. ■

One implication of the above result is that for any Boolean constraint satisfaction problem for which 1) there exists a satisfying assignment, and 2) for every subset of r variables the constraints forbid at least one of

[1] We thank an anonymous reviewer of an early version of this paper for drawing our attention to this.

the 2^r possible assignments, it must be the case that there are only a *constant* number of "ϵ-similar solution clusters," where an ϵ-similar solution cluster is a set of assignments that differ from each other in at most ϵn locations. Indeed, the number of such clusters will be at most 2^k, where $k = \tilde{O}(r/\epsilon)$ is as specified in Theorem 3 and Proposition 4, is a bound on the number of variables whose assigned value must be queried to achieve a constant probability of failure $\delta < 1$. Note that this number of solution clusters is independent of n.

This structure of the satisfying assignments is slightly surprising given the following two simple examples: the first example illustrates that it is possible for such CSPs to have at least two extremely different satisfying assignments, and the second illustrates that it is possible for such CSPs to have super-constant sized solution clusters—clusters of size $\Omega(n)$—although all the assignments in such a cluster are quite similar.

Example 1 *Consider the setting where the underlying assignment to all n variables is T, and for every pair of variables, the set of allowable assignments is $\{(F,F),(T,T)\}$. Based on these constraints, there are two possible satisfying assignments—either all T or all F. A single "verified" data point is sufficient to distinguish between these two sets of assignments.*

The following example illustrates that, in general, it is impossible to guarantee that the learner will correctly output the *exact* assignment, unless the number of verified datapoints $k = \Theta(n)$.

Example 2 *Consider the setting where each set of r values has the constraint that precludes the $(F,F,\ldots,F)$ r-tuple. In this case, there is a single solution cluster consisting of all assignments to the n variables such that at most $r-1$ of the variables are F and the remaining $n-r+1$ are T. In this case, it is impossible to distinguish between these assignments with any significant probability using fewer than $\Theta(n)$ verified evaluations.*

Despite the above examples, it is still unclear whether the information theoretic bound of Proposition 4 is tight; particularly for small constant ϵ, it is not clear the extent to which the number of ϵ-separated solution clusters can grow as ϵ decreases. This seems like an intriguing and fundamental question related to the "semi-adversarial" setting of Proposition 4 where the choice of forbidden assignments is chosen adversarially subject to being consistent with a planted assignment.

1.3. Related Work

The general challenge of developing algorithms and estimators that are robust to corruptions in the input data dates back to the early work of Tukey (Tukey, 1960), and has led to a significant body of work on "Robust Statistics", which explores a number of different models for the data corruptions, and largely focuses on the regime in which a majority of the data is "good." We refer the reader to the surveys of (Huber, 2011) and (Hampel et al., 2011) for overviews of this literature.

Motivated by the increasing practical importance of robust estimation—and more generally, robust learning and optimization—there has been recent interest in these problems from both an information theoretic and computational perspective. Recent works from the TCS community tackled this general problem in several basic settings, including robustly estimating the mean and covariance of natural classes of distribution, including multivariate Gaussians (Diakonikolas et al., 2016; Lai et al., 2016). The focus of these works was largely on establishing computationally efficiency algorithms for these tasks that approach the information theoretic (minimax) guarantees achieved by more naive or brute-force algorithms. There has been a flurry of very recent works both tightening these results and extending them to more general classes of distribution and other optimization or learning problems (Steinhardt et al., 2018; Kothari and Steinhardt, 2017; Kothari and Steurer, 2017; Diakonikolas et al., 2017a). This work differs significantly from the long line of work on agnostic learning, in that here the adversary is allowed to corrupt both the data distribution, and the labels, in contrast to the more limited agnostic learning setting where only the labels are corrupted.

The above works focus on the setting where a significant majority of data is "good". The works (Steinhardt et al., 2016) and (Charikar et al., 2017) and the very recent work (Diakonikolas et al., 2017b) consider the setting where a minority of the data is "good" (i.e. $\alpha < 1/2$), with (Charikar et al., 2017) formally proposing the "semi-verified" learning model where one may obtain a small amount of "verified" data that has been drawn from the distribution/cohort in question. The work (Steinhardt et al., 2016) considers a similar item evaluation setting to the setting we consider, but focuses on the regime where the number of evaluators is on the same order as the number of items being evaluated and the r-tuples of items being evaluated are selected uniformly at random. In this regime, they show that ϵ-accurate recovery is possible provided that the number of items reviewed by each evaluator is $O(\frac{1}{\epsilon^4 \alpha^3})$.

In contrast, we consider the regime in which the number of evaluators might be significantly larger than the number of items, or the items to evaluate are not chosen at random, but establish an optimal tradeoff between the fraction of good reviewers and the number of items evaluated by each reviewer, demonstrating the surprising ability to tolerate a fraction of good reviewers that is inverse exponential in the number of items evaluated by each reviewer. For the context of leveraging these techniques as a "prism" to extract information about specific demographics from a large, mixed dataset, this small-α regime seems especially significant. The techniques of this paper, via local algorithms and the constraint-satisfaction perspective, also differ significantly from the previous approaches to robust estimation which rely on more geometric or spectral structure.

2. The Algorithm Intuition

In this section we describe the intuition and high-level structure for the algorithm to which Theorems 1, 2 and 3 applies. Throughout, it will be convenient to assume that we have constraints on sets of $r = 1 + \lfloor \log_{2-2p}(1/\alpha) \rfloor$ variables. This is without loss of generality, since provided $r \geq 1 + \lfloor \log_{2-2p}(1/\alpha) \rfloor$, we could consider only the implied constraints on subsets of exactly $1 + \lfloor \log_{2-2p}(1/\alpha) \rfloor$ variables.

The overall structure of the algorithm is to iteratively reduce an instance of the problem with non-trivial constraints on sets of r variables, to an instance of the problem that has non-trivial constraints on sets of $r-1$ variables. In general, the true assignment might *not* satisfy the constraints that we derive on the sets of $r-1$ variables, though we will be able to leverage any such derived constraints that are discovered to be false.

We describe the intuition for the algorithm in two parts. First, in Section 2.1 we describe an algorithm with runtime $O(n^2)$ for the case that $r = 2$. We then sketch how to convert this algorithm into a linear-time algorithm in Section 2.2. In Section 2.3 we sketch the intuition for the reduction from constraints on sets of r variables to constraints on $r-1$-tuples. We formally describe the general linear-time algorithm and provide its proof of correctness in Appendix A; this algorithm combines both the constraint reduction idea of Section 2.3 with the machinery for making the algorithm run in linear time described in Section 2.2.

2.1. Intuition: Restricting to Pessimistic Constraints

Our algorithm will proceed iteratively, with the goal of each iteration being to inspect at most a constant number of randomly sampled "verified" variable values, and return accurate guesses for at least a constant fraction of the variables. The algorithm will then recursively iterate this procedure on the remaining variables until all but $\frac{\epsilon}{2}n$ variables have been assigned guesses; assignments to these last $\leq \epsilon n/2$ variables can be chosen arbitrarily.

To begin, consider the setting where $r=2$, and for every pair of variables (x,y) we have a set of allowable assignments, $C_{(x,y)} \subset \{T,F\}^2$, with $|C_{(x,y)}| \leq 3$. Each such set provides at least two implications, one of the form $x = X \implies y = Y$ and one of the form $y = Y' \rightarrow x = X'$ for some choice of $X, X', Y, Y' \in \{T,F\}$. For example, if the assignment $(T,F) \notin C_{(x,y)}$, then we have the implications $x = T \implies Y = T$ and $y = F \implies x = F$. In other words, there is at least one value of variable x that would imply the value of variable y, and similarly for y.

Hence, if we fix variable x, and consider the implications derived from the sets $C_{(x,y)}$ as y ranges over all $n-1$ other variables, there must be an assignment to variable x that would imply the values of at least $n/2$ variables. We will refer to this assignment as the "optimistic" value of x, as this assignment to x would immediately yield the values of at least half the remaining variables, and we would be done with the current iteration of the algorithm, and would then recurse on the remaining $< n/2$ variables that have not been assigned values.

The first key idea of our algorithm is that we will assume that *all* variables take their "pessimistic" values. We will then "check" this assumption by revealing the true values of a random sample of $O(\log(1/\delta)/\epsilon)$ of these variables. If all of these values are consistent with the "pessimistic" values, we can conclude that with probability at least $1-\delta$, at least $(1-\epsilon)n$ of the variables actually take their "pessimistic" values, and hence we can simply output this assignment. If, however, any of our $O(\log(1/\delta)/\epsilon)$ random checks fails, that means that we have found a variable that takes its "optimistic" value, and hence that one variable, x, together with the $n-1$ constraint sets $C_{(x,\cdot)}$ that involve it, imply the values of at least $n/2$ variables. In either case, our constant (dependent on ϵ, δ) number of checks has yielded an accurate assignment to at least half the variables. This simple algorithm in the $r=2$ case is summarized in the following pseudo-code:

```
FindAssigments, $r=2$:
Input: Set of $n$ variables, and for every pair $(x,y)$, a set of allowable assignments to those variables $C_{(x,y)} \subset \{T,F\}^2$, with $|C_t| \leq 3$. Error parameter $\epsilon > 0$ and failure parameter $\delta > 0$.
Output: Assignments to each of the $n$ variables.

• While there exists $\geq \epsilon n/2$ variables without assignments
    – Let $n'$ denote the number of remaining variables, and for each of these, determine an "optimistic" assignment that would imply the values of at least $n'/2$ other variables, and define a variable's "pessimistic" value to be the opposite assignment.
    – Consider a set of $10\frac{\log(1/\delta)}{\epsilon^2}$ randomly chosen variables and their "verified" assignments. (If fewer than $\frac{\log(1/\delta)}{\epsilon}$ of these variables lie in the set of $n' > \epsilon n/2$ variables in consideration, output FAIL)
    – If all the verified assignments for variables in the set of $n'$ agree with their pessimistic assignments, then assign these $n'$ variables their pessimistic assignments.
    – Otherwise, we must have found a variable whose verified assignment is its *optimistic* assignment, and we can assign the values of at least $n'/2$ variables accordingly.
```

2.2. Sketch of Linear Runtime

The above algorithm, as presented, would require a quadratic runtime, as determining the "optimistic" assignment to each variable, x, (the assignment that would imply the values for at least half the variables) requires inspecting a linear number of constraints—namely the constraints $C_{(x,x')}$ as x' ranges over the $O(n)$ variables. This, however, can easily be avoided via a sampling based approach: for each variable, x, it is sufficient to examine a constant, dependent on ϵ and δ, number of constraints, to determine an assignment that, with good probability, would imply the values of at least a 0.49 fraction of the remaining variables.

Given (mostly) accurate assignments of "optimistic" values to the variables, the algorithm of the previous section proceeds by getting verified values for a constant-sized set of k of these variables. If none of them agree with the optimistic assignments, then we conclude that most variables take their pessimistic values, and we are done in linear time, via the constant-query sampling to determine the optimistic/pessimistic values of each of the $O(n)$ variables.

If, however, one of the k verified variables takes its optimistic value, at an additional $O(n)$ cost, we explore the $O(n)$ implications implied by this optimistic assignment; this additional step also clearly takes linear time.

Finally, note that we do not need to know the verified values for the k variables before querying the implications that would be implied if each of them were to take their optimistic values. Hence, given only the identities of the k verified variables (but not their values), at a cost of $O(kn) = O(n)$ we could probe a linear number of constraints, such that after the values of the k variables are revealed, we can compute the assignment to all $> 0.49n$ variables using only the linear amount of information and linear computation. In the crowd-sourcing setting, this corresponds to collecting $\tilde{O}(kn)$ evaluations of pairs of items, $(x_1,\cdot),\ldots,(x_k,\cdot)$ where $x_1,\ldots,x_k$ represent the k variables for which we will receive verified evaluations; based on this linear amount of information, at a later time, one could collect the verified evaluations of the prescribed k items, and then output accurate assignments to the items.

2.3. From r-tuples to $r-1$-tuples: Pessimism All The Way Down

Given the algorithm for the $r = 2$ case, which is successful provided every pair of variables has at least one forbidden assignment, the question is how to reduce the setting with constraints on sets of $r \geq 3$ variables, to the setting of constraints on sets of $r-1$ variables. The following observation is the key to this reduction:

Lemma 5 *Given an r-tuple and set of at most $2^r - 1$ allowable assignments to those r variables, then for any subset of $r-1$ of those variables, there exists an assignment to those $r-1$ variables that would imply the value of the rth variable.*

Proof Consider an $r-1$ tuple, t, and an additional variable v, and the set of $\leq 2^r - 1$ allowable assignments to the r-tuple $(t \cup v)$. If the restriction of these assignments to the $r-1$ variables in t contains all 2^{r-1} possible assignments, it must be the case that for at least one of these assignments, there is a unique value that v must assume, otherwise this r-tuple would have all 2^r possible assignments. If the restriction of the $2^r - 1$ assignments to the $r-1$ tuple does not contain all 2^{r-1} assignments, then any assignment $\sigma \in \{T,F\}^{r-1}$ that cannot be obtained as a restriction of one of the assignments to the $r-1$ variables, vacuously implies the value of the rth variable. ■

The utility of this lemma is that if we have an $r-1$-tuple of variables, t, then by considering all possible additional variables $v \notin t$, there exists an assignment to t that determines the value of at least a $1/2^{r-1}$ fraction of the variables not in t. Hence we can designate an "optimistic" assignment with the property that if that assignment holds, then it will imply assignments to at least a $1/2^{r-1}$ fraction of the remaining variables. We will then assume that this "optimistic" assignment is *not* allowed, thereby reducing the set of allowable assignments of variables in t to size $2^{r-1} - 1$, and proceed inductively. In this sense, at some intermediate step of this algorithm where we are considering sets of $r' < r$ variables, the allowable sets of assignments that we are considering may not be completely accurate, as we are not verifying whether the sets actually do take their "optimistic" assignments or not. However, if a r'-tuple of variables actually takes the values of a forbidden/optimistic assignment, then either it will immediately imply the values of a constant (i.e. at least $1/2^r$) fraction of variables, or it must be a subset of a larger tuple that takes its "optimistic" assignment. Which of these two cases holds can be easily decided via querying the values of a (constant) number of random variables. We describe the full algorithm in the following section.

2.4. The Inefficient Algorithm

The high-level structure of the algorithm described in the previous part takes the form of a "descending" pass followed by an "ascending" pass. In the descending pass, we iteratively turn constraints on r_0 tuples into constraints on $r_0 - 1$ tuples, then $r_0 - 2$ tuples, etc; all the while, we forbid "optimistic" assignments to ensure that in the rth level, each r tuple has at most $2^r - 1$ allowable assignments. This descending phase terminates with $r = 1$, where we have our "pessimistic" conjectured assignments to all variables. We then randomly check a few of these values; if we do not discover any inconsistencies with the conjectured values, then we can safely conclude that most of the conjectured values are correct.

If we have discovered any inconsistencies, then we begin the ascending phase that investigates and checks any discovered "optimistic" assignments. One minor wrinkle is that we should not trust the $> 1/2^r$ fraction of values that appear to be implied by an optimistic assignment to a set of $r < r_0 - 1$ variables. These implications might be the result of forbidding an optimistic assignment for some larger tuple. Nevertheless, if we randomly check some of the implications, then we will either verify the accuracy of these implications, or have found an optimistic assignment to a $r + 1$ tuple. In this sense, the ascending phase will either terminate upon satisfactorily verifying a significant (constant sized) subset of the set of output assignments, or we will have found an "optimistic" assignment to a $r_0 - 1$ tuple, and the implications of $r_0 - 1$ tuples are based directly on the given set of constraints to r_0-tuples, which are valid by assumption. Hence each phase of the algorithm will return assignments to a constant (at least $1/2^{r_0}$) fraction of the variables.

We formally describe this algorithm now. As described, the runtime of is not linear in the number of items, n, and this algorithm will require a number of "verified" samples that is inverse polynomial in the error parameter ϵ, as opposed to the nearly inverse linear dependence specified in the Theorem 3. The algorithm to which Theorem 3 applies is an extension of this algorithm, and results from combining the ideas of Section 2.2 with this algorithm. The full linear-time variant of this algorithm is provided in Appendix A.

FindAssigments:
Input: Set of n variables, integer r_0, and for every tuple $t \subset [n]^{r_0}$ of r_0 distinct variables, a set of allowable assignments to those variables $C_t \subset \{T, F\}^{r_0}$, with $|C_t| \leq 2^{r_0} - 1$. Error parameter $\epsilon > 0$ and failure parameter $\delta > 0$.
Output: Assignments to at least $(1 - \frac{\epsilon}{2})n$ variables.

- While there exists $\geq \epsilon n/2$ variables without assignments, run DESCEND on the set of unassigned variables and their corresponding sets of allowable assignments.

```
DESCEND:
Input: For each $r$-tuple, a set $C$ of assignments to those variables, with $|C| \leq 2^r - 1$.
  1. If $r = 1$, AscendAndVerify(set of assigned values to each variable, $r = 1$).
  2. Else, for every $r-1$ tuple, $t$, we will create a set $C_t$ of $\leq 2^{r-1} - 1$ assignments:
       • Find an "optimistic" assignment $\sigma_t$ that would determine at least a $1/2^r$ fraction of variables not in $t$. (The existence of such an assignment is guaranteed by Lemma 5.)
       • Set $C_t = \{T,F\}^{r-1} \setminus \sigma_t$.
  3. Run DESCEND on the set of $r-1$-tuples and their corresponding sets of assignments, each of size $2^{r-1} - 1$.
```

```
ASCEND AND VERIFY:
Input: Proposed assignments $\sigma_v$ for each variable $v \in V$ for some set $V$ of variables. Integer $r$ indicating the size of the tuples whose constraints generated the proposed assignments, and assignment $\sigma_t$ to a $r-1$-tuple $t$, such that $C_{t\cup v}$ provided the implication $\sigma_t \rightarrow \sigma_v$. Access to sets of allowable assignments corresponding to all tuples of size $r' \in \{r,...,r_0\}$. Constant $A = 2^{r_0}\log(1/\delta)\log(1/\epsilon))/\epsilon^2$.
  1. Randomly sample $A$ verified variable assignments.
  2. If all verified variable assignments agree with the proposed assignments, $\sigma_v$, then permanently assign $v \in V$ with their proposed assignments, $\sigma_v$.
  3. Otherwise, let $v$ denote a variable whose true/verified assignment $a_v \neq \sigma_v$, disagrees with the proposed assignment to $v$. Hence $(\sigma_t, a_v) \notin C_{t\cup v}$ so assignment $(\sigma_t, a_v)$ together with the constraints on the $r+1$ tuples must imply at least a $1/2^r$ fraction of variable assignments. Denote these assignments by $\sigma^{new}$.
       • Run AscendAndVerify($\sigma^{new}$, $r+1$, $\{t \cup v\}$, $(\sigma_t, a_v)$)
```

3. Future Work

This work shows that it is possible to tolerate a fraction of "good" data, α, that is inverse exponential in r_0, the sparsity of each datapoint (i.e. the number of evaluations submitted per reviewer), provided the number of datapoints/reviewers is sufficiently large to ensure that each set of r items queried has been evaluated by a significant number of good reviewers. Our algorithm runs in time linear in the number of items to review (provided the ability to query summary statistics of the set of reviewers who have evaluated a given sets of items), and uses a constant number of "verified" reviews, which is independent of the total number of items to review, and depends inverse linearly on the desired error (to logarithmic factors).

One natural question is prompted by the results of (Steinhardt et al., 2016), which provide efficient algorithms for the regime where $r = poly(1/\alpha)$, but where it suffices to have a linear number of reviewers that independently choose sets of r items to evaluate. (In our setting, we would require $> n^r$ reviewers if each chooses their set uniformly at random—the setting where we have positive results using $\tilde{O}(n)$ reviewers requires that we can choose the sets of r items to assign to each reviewer.) Is it possible to achieve the best-of-both-worlds: $r = polylog(1/\alpha)$, and a number of reviewers that is linear, or grows significantly more slowly than the n^r that we require, while leveraging a constant number of verified reviews?

To this end, our algorithm only ever considers "single-hop" implications of proposed assignments: an assignment to a set of r variables is considered "optimistic" if it directly implies values for a significant fraction of the other variables. It is easy to imagine extending this definition to also consider longer chains of implication. Perhaps a specific assignment to r variables would imply values to c_1 additional variables, which in turn would imply values to c_2 variables, etc. Indeed, in the basic setting of $r = 2$, this approach can be realized to yield an algorithm that only requires constraints on a random subset of size $O(n^{3/2})$, as opposed to $O(n^2)$ constraints.

From a computational perspective, it seems unlikely that such an approach could be pushed to yield an efficient algorithm for the regime in which fewer than $n^{r/2}$ sets of r variables have nontrivial constraints. Indeed, even for random instances of $r - SAT$ with a planted solution, efficient algorithms below this threshold have been elusive (see, for example, the recent related work on random CSPs with planted assignments (Feldman et al., 2015; Raghavendra et al., 2016)).

From a purely information theoretic perspective—the picture is not entirely clear either. In contrast to random CSPs with planted assignments, for which constraints are placed on random r-tuples and the constraints are chosen randomly subject to respecting the planted assignment, our setting is complicated by the adversarial nature of the constraints that are placed on the r-tuples. In a fully adversarial CSP model, for which both the choices of the r-tuples as well as the constraints themselves are chosen adversarially—to the best of our knowledge—very little is known. Of course, in this setting, the goal is to find a satisfying assignment (that might not necessarily correspond to the planted assignment). In the semi-adversarial CSP model, where the identities of the r-tuple sets are picked at random, and the adversary chooses the constraints, our results show that we can recover an assignment, provided we can selectively query $O(n)$ of the $\binom{n}{r}$ possible constraints. In these settings it is not immediately clear how to analyze the extent to which implications "propagate". A second difficulty is that the goal of our setting is not just to find a satisfying assignment, but to find something close to a specific planted assignment. Our results imply, for the settings we consider, that there are at most a constant number of solution clusters. It seems interesting to investigate the extent to which this holds for semi-adversarial CSPs with fewer constraints, perhaps with constraints chosen adversarially corresponding to only $N \ll \binom{n}{r}$ random r-tuples; in this setting it seems plausible that $N = n^{r/2}$ is the threshold between a constant and super-constant number of such solution clusters, though this might be difficult to prove.

References

Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6): 925–936, 2010.

M. Charikar, J. Steinhardt, and G. Valiant. Learning from untrusted data. In *Symposium on Theory of Computing (to appear)*, 2017.

Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 655–664. IEEE, 2016.

Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Learning geometric concepts with nasty noise. *arXiv preprint arXiv:1707.01242*, 2017a.

Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. List-decodable robust mean estimation and learning mixtures of spherical gaussians. *arXiv preprint arXiv:1711.07211*, 2017b.

Vitaly Feldman, Will Perkins, and Santosh Vempala. On the complexity of random satisfiability problems with planted solutions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 77–86. ACM, 2015.

Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*, volume 114. John Wiley & Sons, 2011.

David Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded vapnik-chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69(2):217–232, 1995.

Peter J Huber. *Robust statistics*. Springer, 2011.

Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2010.

Pravesh K Kothari and Jacob Steinhardt. Better agnostic clustering via relaxed tensor norms. *arXiv preprint arXiv:1711.07465*, 2017.

Pravesh K Kothari and David Steurer. Outlier-robust moment-estimation via sum-of-squares. *arXiv preprint arXiv:1711.11581*, 2017.

Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 665–674. IEEE, 2016.

Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random csps below the spectral threshold. *arXiv preprint arXiv:1605.00058*, 2016.

Jacob Steinhardt, Gregory Valiant, and Moses Charikar. Avoiding imposters and delinquents: Adversarial crowdsourcing and peer prediction. In *Advances in Neural Information Processing Systems*, pages 4439–4447, 2016.

Jacob Steinhardt, Moses Charikar, and Gregory Valiant. Resilience: A criterion for learning in the presence of arbitrary outliers. In *Innovations in Theoretical Computer Science (ITCS)*, 2018.

John W Tukey. A survey of sampling from contaminated distributions. *Contributions to probability and statistics*, 2:448–485, 1960.

Appendix A. An Efficient Algorithm

The linear-time variant of the basic algorithm described in Section 2.4 leverages the two ideas sketched in Section 2.2 for the $r=2$ setting. We now formalize them in this general r setting. For a given r-tuple t, rather than consulting all $\Theta(n)$ constraints $C_{t\cup x_i}$ for all $x_i \notin t$ to determine the "optimistic" assignment to t, one can determine an assignment that implies at least a $\frac{1}{2}\frac{1}{2^r}$ fraction of the variable values, with high probability, via *sampling* a constant (independent of n but dependent on r,ϵ,δ) number of such constraints. Note that this sampling does not look at any of the "verified" variable assignments—it just samples which of the constraints to consider. We formalize this ability to efficiently determine an "optimistic" assignment via the following subroutine, and the following lemma characterizing its performance.

```
FIND OPTIMISTIC ASSIGNMENT:
Input: Set of $n$ variables $X$, $r$-tuple $t$, the ability to query constraints $C_{t'}$ for $|t'| = r_0 \geq r$ (i.e. the ability to find optimistic assignments to tuples $t'$ with $|t'| = r_0$) and probability of failure $\gamma > 0$.
Output: An optimistic assignment $\sigma_t$ to $t$ that would, with probability at least $1-\gamma$, imply the assignments to at least a $1/2^{r+1}$ fraction of other variables via the constraints $C_{t \cup x}$. We define $C_t := \{T,F\}^r \setminus \{\sigma_t\}$.

  1. If $r = r_0$ then return constraint $C_t$.
  2. Else
       • Select $s = 3 \cdot 2^{|t|} \log(1/\gamma)$ variables $x_1,...,x_s$ uniformly at random from $X \setminus t$.
       • For each of these $s$ variables, $x_i$, compute $C_{t \cup x_i}$ via a (recursive) call to $FindOptimisticAssignment(X, t \cup x_i, ProbFailure = \gamma/(2s))$.
       • Define assignment $\sigma_t \in \{T,F\}^r$ to be the lexicographically first assignment that, via the constraints $\{C_{t \cup x_i}\}$, implies at least a $1/2^r$ fraction of variables $\{x_1,...,x_s\}$. [Note that such an assignment exists, since for each $x_i$, $|C_{t \cup x_i}| \leq 2^{r+1}-1$ has at least one out of the $2^r$ possible assignment to $t$ that would imply that value of $x_i$.]
       • Call $\sigma_t$ the "optimistic" assignment to tuple $t$, and store $C_t = \{T,F\}^r \setminus \sigma_t$.
```

The following two lemmas quantify the performance of the above algorithm. The first lemma characterizes the probability of failure, and the proof follows immediately from standard Chernoff tail bounds.

Lemma 6 *With probability at least $1-\gamma$ the optimistic assignment σ_t returned by algorithm FindOptimisticAssignment on input X and t has the property that for at least a $1/2^{|t|+1}$ fraction of variables $x \in X$, the assignment σ_t together with the constraint set $C_{t \cup x}$ that would be computed by the algorithm on input tuple $t \cup x$, implies the value of variable x.*

Proof Letting p denote the true fraction of variables, x, whose assignments are implied by σ_t and $C_{t \cup x}$. Recall that σ_t was chosen based on s independent samples, yielding an empirical estimate $\hat{p} \geq 1/2^{|t|}$, and standard tail bounds yield that $\Pr[\hat{p} > 2p] \leq e^{-\frac{s}{3 \cdot 2^{|t|}}}$, yielding the lemma, since $s = 3 \cdot 2^{|t|} \log(1/\gamma)$. ■

Lemma 7 *Given constant-time query access to the constraint sets $C_{t'}$ for tuples satisfying $|t'| = r_0$, for any tuple t, algorithm FindOptimisticAssignment on input t and probability of failure $\gamma > 0$ returns C_t and runs in time/queries $(2\log(1/\gamma))^{O(r_0^2)}$, which is independent of the size of the variable set, $|X|$.*

Proof Note that computing C_t calls $O(2^{|t|}\log(1/\gamma))$ computations of $C_{t'}$ for $|t'| = |t|+1$, each called with error parameter $2^{|t|}$ smaller. When $|t| = r_0$, C_t is obtained via a single constant-time query. Expanding this recursion yields the above lemma. ■

The second observation that underpins the efficient algorithm is that we do not need to determine the optimistic assignments and form constraints C_t for all $\binom{n}{r}$ r-tuples t. For each phase of the algorithm, which returns assignments to a constant fraction of the unassigned variables—at least $1/2^{r_0}$—it suffices to find a single tuple t that takes its "optimistic" assignment. Indeed, such a tuple, by definition, takes values that imply assignments to a constant fraction of the remaining variables. And for each of these variables,

x, whose assignment is implied by the assignment to the tuple t, the value of variable x can be determined in constant time by consulting the constraint $C_{t\cup x}$. This observation is clarified in the following algorithm, which is an adaptation of the Descend/AscendAndVerify algorithm described in the previous section.

Finally, we highlight the fact that the algorithm proceeds iteratively. Given an initial set of variables, Y, at some intermediate step in the algorithm, we let X denote the set of variables for which we have not yet output an assignment. The algorithm will terminate when $|X| \leq \epsilon|Y|/2$. The goal of the current step of the algorithm will be to output assignments to at least a $1/2^{r_0}$ fraction of variables in X, such that the fraction of such assignments that are incorrect is bounded by $\frac{\epsilon}{2\log(2/\epsilon)}\frac{|Y|}{|X|}$. Given this bound on the fraction of incorrect assignments returned at this phase of the algorithm, the total fraction of errors is bounded by $\epsilon/2+\int_{t=\epsilon/2}^{1}\frac{\epsilon}{2\log(2/\epsilon)}\cdot\frac{1}{t}dt=\epsilon$ where the first $\epsilon/2$ is a bound on the error due to the arbitrary assignments to the last $\leq \epsilon|Y|/2$ variables. The benefit of having the target accuracy increase as $|X|$ decreases is because we are given verified samples, drawn uniformly at random from Y. To "check" a proposed assignment to set X to a target accuracy of γ, we need at least $1/\gamma$ verified samples from the set X (ignoring the logarithmic dependence on the probability of failure). To guarantee that this number of verified samples is obtained from set X, we will need to draw $\approx \frac{|Y|}{\gamma|X|}$ verified samples from Y. Using the above trick of having the desired accuracy degrade as $|X|$ decreases, for each phase of the algorithm, a set of $\frac{|Y|/|X|}{\frac{\epsilon}{2\log(2/\epsilon)}\frac{|Y|}{|X|}}=\tilde{O}(1/\epsilon)$ verified samples is required—as opposed to the $\Theta(1/\epsilon^2)$ samples that would have been required if we had fixed the target error rate to be ϵ for all rounds of the algorithm.

```
EFFICIENT FIND ASSIGNMENTS:
Input: Set of $n$ variables $Y$, integer $r_0$ and for every $r_0$-tuple $t \subset Y$, a set of allowable assignments $C_t$ with $|C_t| \leq 2^{r_0}-1$. Error parameter $\epsilon>0$ and probability of failure $\delta$.
Output: Set of $T/F$ assignments to each $x \in Y$.

  • Set $T = r_0 \cdot 2^{r_0+1}\log(2/\epsilon)$.
  • While there are at least $\epsilon n/2$ unassigned variables:
      1. Let $X \subset Y$ denote the set of unassigned variables.
      2. Let $\epsilon_X = \frac{\epsilon}{2\log(2/\epsilon)}\frac{|Y|}{|X|}$ denote the target accuracy of this round, and set $s = 10\frac{|Y|}{\epsilon_X|X|}\log(10T/\delta)$.
      3. Take $s$ verified samples, revealing the planted assignment values for each of these variables.
         Let $X_1 \subset X$ denote the subset of these variables that are in set $X$, and for each $x \in X_1$ let
         $a_x$ denote the verified assignment to variable $x$. If $|X_1| < s\frac{|X|}{2|Y|}$ output FAIL.
      4. For each $x \in X_1$, determine $C_x$ via FindOptimisticAssignments with failure parameter
         $\gamma = \delta/T$.
      5. If, for all $x \in X_1$, $a_x = C_x$, then for every variable $x \in X$, compute and output assignment $C_x$.
      6. Otherwise, let $x_1 \in X_1$ denote a variable for which $a_x \neq C_x$, and run
         EfficientAscend($X$,$i$,$x_1$,$a_{x_1}$,$s$).
```

EFFICIENT ASCEND:
Input: Set of variables X, integer $i \in \{1,...,r_0-1\}$, tuple t with $|t|=i$, verified assignments $a_t \in \{T,F\}^{|t|}$ to tuple t, and parameter s.
Output: Output to a subset of variables in set X.

1. If $i \geq r_0$ output FAIL.

2. Take $s_i = s \cdot 2^i$ verified samples, and let X_{i+1} denote the intersection of X with this set of variables with verified assignments, with a_x denoting the verified assignment to variable $x \in X_{i+1}$.

3. For each $x \in X_{i+1}$, determine $C_{t \cup x}$ via a call to FindOptimisticAssignment(X, $t \cup x, FailureProb = \delta/(10T \cdot s_i)$), and let $X'_{i+1} \subset X_{i+1}$ denote the subset of variables $x \in X_{i+1}$ for which the constraint $C_{t \cup x}$ together with a_t implies a value σ_x for x. If $|X'_{i+1}| \leq s_i \frac{|X|}{4 \cdot 2^i |Y|}$ output FAIL.

4. If, for all $x \in X'_{i+1}$, it holds that $\sigma_x = a_x$, then for every variable $x \in X$, compute $C_{t \cup x}$ and output assignment σ_x if σ_x is implied by $C_{t \cup x}$ and a_t.

5. Otherwise, let $x_{i+1} \in X'_{i+1}$ denote a variable for which $C_{t \cup x_{i+1}}$ and a_t implies assignment $\sigma_{x_{i+1}} \neq a_{x_{i+1}}$. Run EfficientAscend($X, i+1, t \cup x_{i+1}, (a_t, a_{x_{i+1}}), s$).

Proposition 8 *Algorithm EfficientFindAssignments, when run with error parameter ϵ and probability of failure δ, has the following properties:*

- *The algorithm will require at most $\tilde{O}\left(\frac{1}{\epsilon} \cdot 2^{2r_0} \log(1/\delta)\right)$ verified samples drawn uniformly at random from the set of variables, Y.*

- *With probability at least $1-\delta$, the algorithm will output assignments to each variable $x \in Y$, such that at most an ϵ fraction of the assignments disagree with the planted assignment.*

- *The algorithm runs in time $O_{r_0,\epsilon,\delta}(n)$, and only consults this number of constraints, $C_{(\cdot)}$.*

Proof The high level outline of the execution of algorithm EfficientFindAssignments is that in each step of the outer WHILE loop, an assignment to at least a $1/2^{r_0+1}$ fraction of the remaining unassigned variables, X, will be output. This continues until $|X| \leq \epsilon n/2$, at which point these remaining variables can be assigned arbitrary labels and the algorithm terminates. Hence there will be at most $O(2^{r_0} \log(1/\epsilon))$ iterations of the while loop. In the iteration conducted on unassigned variable set $X \subset Y$, the goal will be to return assignments such that the fraction of returned assignments that are incorrect is at most $\frac{\epsilon}{2\log(2/\epsilon)} \frac{n}{|X|}$, where $|Y| = n$ is the total number of initial variables. Provided these accuracy goals are met at each step of the algorithm, the overall fraction of errors will be bounded by $\epsilon/2 + \int_{f=\epsilon/2}^{1} \frac{1}{f} \cdot \frac{\epsilon}{2\log(2/eps)} df = \epsilon$, where the first term is the errors due to the arbitrary assignment to the remaining $\leq \epsilon n/2$ variables. Additionally, the number of verified samples required in each iteration is at most $O(r_0 s \cdot s 2^{r_0}) = \tilde{O}\left(2^{r_0} \frac{1}{\epsilon} \log(1/\delta)\right)$, hence the total number of verified samples across the $O\left(2^{r_0} \log(1/\epsilon)\right)$ iterations will be bounded by $\tilde{O}(2^{2r_0} \log(1/\delta)/\epsilon)$, as claimed.

We now analyze each run of the WHILE loop in EfficientFindAssignments, and the recursive calls to EfficientAscend. At a high level, in each recursive call to EfficientAscend, either an assignment to at least a

$1/2^{r_0+1}$ fraction of the remaining unassigned variables is returned via the implications from some (verified) optimistic assignment to a tuple, t; or, we have found a tuple $t \cup x_{i+1}$ for which we have verified assignments to all $|t|+1$ variables, and for which that assignment, $(a_t, a_{x_{i+1}}) \notin C_{t \cup x_{i+1}}$ is the optimistic assignment, in which case the subsequent call to EfficientAscend considers this strictly larger tuple $t' = t \cup x_{i+1}$.

To bound the runtime of the algorithm, note that each run of the algorithm requires constant time (dependent on r_0, ϵ, δ but independent of the number of variables, $|X|$, up until the point in the algorithm when an assignment will be output (Step 4 in EfficientAscend). At this point in the algorithm, at a computational expense of $O_{r_0,\epsilon,\delta}(|X|)$, an assignment to a constant fraction, at least $1/2^{r_0+1}$ of the remaining variables will be output, and the algorithm will then be repeated on the remaining unassigned variables. Hence, the overall runtime of the algorithm will be linear in the number of variables.

To bound the probability that a given run of the WHILE loop fails to successfully output an assignment to at least $|X|/2^{r_0+1}$ variables that meets the target accuracy of $\frac{\epsilon}{2\log(2/\epsilon)} \frac{|Y|}{|X|}$, we will leverage a union bound over a number of standard Chernoff tail bounds. First, note that the probability that EfficientFindAssignments outputs 'FAIL' in Step 3 in a given round of the algorithm is bounded by the probability that $|X_1| \leq \mathbf{E}[|X_1|/2]$, where X_1 is a sum of i.i.d 0/1 random variables, hence this probability is bounded by $exp(-E[|X_1|]/8) \leq \frac{\delta}{10T}$, where T, as specified in EfficientFindAssignments is a bound on the number of calls to EfficientAscend which bounds the number of runs of the WHILE loop. Given that $|X_1| \geq \mathbf{E}[|X_1|/2]$, the probability that the assignment output in Step 5 of EfficientFindAssignments does not meet the target accuracy, $\epsilon_X = \frac{\epsilon}{2\log(2/\epsilon)} \frac{|Y|}{|X|}$, is bounded by $(1-\epsilon_X)^{|X_1|} \leq \frac{\delta}{10T}$.

The remaining probability of failure stems from the execution of EfficientAscend. In this algorithm, failure can stem from three different issues: 1) the constant number of constraints $C.$ computed via FindOptimisticAssignment prior to Step 4 of EfficientAscend can be erroneous and fail to imply the desired fraction of assignments. The probability of this is bounded by $\delta/(10Ts_i)$, which is sufficient to guarantee that *every* optimistic assignment/constraint set $C.$ that is computed during the execution of the algorithm is accurate and implies the desired fraction of assignments, aside from the $O(|Y|)$ constraints computed during the assignment output steps—Step 4 of EfficientAscend and Step 5 of EfficientFindAssignments. 2) EfficientAscend outputs FAIL during Step 1, though this cannot occur as the constraints corresponding to $i = r_0$ are the constraints on r_0-tuples, which are satisfied by assumption. 3) The final potential failure mode of the algorithm is Step 3 of EfficientAscend, if the random set of verified assignments is insufficiently large to verify (to the target accuracy) a given potential set of assignments implied by an optimistic assignment via $C_{t \cup x}$. Given that the assignment a_t to tuple t is optimistic, as guaranteed by the validity of FindOptimisticAssignments described above, this probability of failure is also a trivial application of standard Chernoff bounds, guaranteeing that the random variable $|X'_{i+1}|$ in Step 3 of EfficientAscend deviates from a lower bound on its expectation by at most a factor of 1/2.

A union bound over these probabilities of failure for each of the $\leq T$ runs of the EfficientAscend algorithm yields the desired proposition. ■