Private Approximation of NP-hard Functions

[Extended Abstract]

Shai Halevi * Robert Krauthgamer † Eyal Kushilevitz ‡ Kobbi Nissim §

ABSTRACT

The notion of *private approximation* was introduced recently by Feigenbaum, Fong, Strauss and Wright. Informally, a private approximation of a function f is another function F that approximates f in the usual sense, but does not yield any information on x other than what can be deduced from $f(x)$. As such, $F(x)$ is useful for private computation of $f(x)$ (assuming that F can be computed more efficiently than f).

In this work we examine the properties and limitations of this new notion. Specifically, we show that for many NP-hard problems, the privacy requirement precludes non-trivial approximation. This is the case even for problems that otherwise admit very good approximation (e.g., problems with PTAS). On the other hand, we show that slightly relaxing the privacy requirement, by means of leaking "just a few bits of information" about x, again permits good approximation.

1. INTRODUCTION

Motivated by problems in secure multiparty computation, Feigenbaum, Fong, Strauss and Wright recently introduced the notion of private approximation [6] (see [7]). In their setting, a set of $n \geq 2$ players wish to compute some value $f(x)$, where the input x is distributed among the players (i.e., $x = (x_1, x_2, \ldots, x_m)$ where x_i is known only to player P_i). As in many other settings, it might be useful for the players to compute some *approximation* for $f(x)$. This happens either because the exact value for f is hard to compute (e.g., if f is an NP-hard optimization problem) or if f is efficiently computable, but its approximation can be computed even more efficiently.

As is the case with the exact evaluation of f, the players want to compute the approximation to f in such a way that will not yield any information about each other's input, other than what can be deduced from the value of $f(x)$ itself. The "obvious solution" would be to apply a secure multiparty evaluation procedure to the approximation algorithm (rather than to the function f itself). Feigenbaum et al. [6, 7] observed, however, that quite often an approximate solution for $f(x)$ reveals some information about x, other than what is implied by the exact solution. Therefore, they defined an approximation to be *private* if no such extra information is revealed. In their work, Feigenbaum et al. focus on "easy" problems for which the exact solution can be efficiently computed, and demonstrate that some of these problems admit extremely efficient (sublinear communication) private approximation. This raises the question of whether private approximation is possible also for "hard" problems, e.g. where finding the exact solution is NP-hard.

We show that for many natural problems, the answer is negative. Namely, for many of the NP-hard approximation problems in the literature, no non-trivial private approximation algorithm exists, unless NP $\subseteq$ BPP. Examples of such problems include minimum vertex cover and MAX-SAT. Moreover, such inapproximability results can be shown even for problems that otherwise admit very good (non-private) approximation. Examples include minimum vertex cover in planar graphs, which admits PTAS and yet do not have non-trivial private approximation algorithms (unless NP $\subseteq$ BPP). We note, however, that so far we do not have examples of problems that admits FPTAS but no non-trivial private approximations.

However, such inapproximability results do not apply to *all* NP-hard problems: We demonstrate how one can concoct an "artificial" problem that would be NP-hard to solve exactly, but for which there exists a private FPTAS. Finding "natural" NP-hard problems that admit non-trivial private approximation is an interesting research direction. Very recently, Feigenbaum et al. [7] obtained a private approximation, that is comparable to the known non-private (ϵ, δ)-approximation, for the *permanent* function.

In light of the above inapproximability results, we turn

*IBM T.J. Watson, P.O. Box 704, Yorktown Heights, NY 10598, USA, Email: shaih@watson.ibm.com

†Department of Computer Science, Weizmann Institute of Science, Rehovot, ISRAEL. Part of this work was done while the author was at the IBM T.J. Watson Research Center. Email: robi@wisdom.weizmann.ac.il

‡Department of Computer Science, Technion, Haifa, ISRAEL. Part of this work was done while the author was at the IBM T.J. Watson Research Center. Email: eyalk@cs.technion.ac.il URL: www.cs.technion.ac.il/~eyalk

§Department of Computer Science, Weizmann Institute of Science, Rehovot, ISRAEL. Part of this work was done while the author was at the AT&T Shannon Labs, NJ. Email: kobbi@wisdom.weizmann.ac.il

STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.

our attention to relaxing the privacy requirement, by means of leaking "just a few bits of information" about the input x. This relaxation is similar to the notion of "additional information" in two-party protocols [4], and to the definition of "knowledge complexity" (in the hint sense) as a relaxation of zero-knowledge (see [11]). We show that by slightly compromising on privacy, we can already get fairly good approximations: Any problem that can be deterministically approximated within ratio α, can also be deterministically approximated within ratio α^2 while leaking only a single bit of information (e.g., there is a 4-approximation for minimum vertex cover that leaks only one bit of information). More generally, leaking at most k bits of information makes it possible to approximate such a problem within ratio $\alpha^{1+1/(2^k-1)}$. We also give some evidence that this ratio may be the best possible for a certain type of problems (and hence in general).

2. PRELIMINARIES

Below we give formal definitions of private approximation, with which we work throughout the rest of this paper. Our definitions follow those of Feigenbaum et al. [6, 7], except for some technical details. These technical differences enable us to simplify the presentation (and notations) of our results but have no significant impact on the results. We discuss these differences in more details in Section 2.1 below. A nice feature of the definitions in [6, 7] is that they separate the *privacy* aspect from the *approximation* aspect. We maintain this feature; we first define what it means for an algorithm F to be private with respect to a function f (Definition 1), then what does it mean for F to approximate f (Definition 2), and finally say that F privately approximates f if it satisfies both definitions.

In the definitions below, $f : X \mapsto \mathbb{N}$ is a deterministic function,[1] and F is a (possibly randomized) algorithm. We denote by $f(x)$ the evaluation of f on x, and by $F(x)$ the output distribution of F on input x.

Informally, an algorithm F is private with respect to a function f, if the output $F(x)$ does not reveal anything about x that cannot be deduced from the value of $f(x)$.[2] This is made formal by insisting that one can efficiently produce a distribution that is "indistinguishable" from $F(x)$, when given only $f(x)$ (and the size of x) as input. The notion of "indistinguishability" used below is essentially the usual notion of computational indistinguishability of distribution ensembles in the uniform model (see, e.g., [9, Sec 3.2]), where the distinguisher sees both x and $f(x)$. We remark that this notion requires some "security parameter". For simplicity, we let the size of x play that role.

Definition 1 (Functional privacy). *Let f be a function and F be an algorithm. The algorithm F is* functionally private *w.r.t. f, if there exists a poly-time "simulator" S such that, for all x, the distributions $\langle x, f(x), S(1^{|x|}, f(x))\rangle$ and $\langle x, f(x), F(x)\rangle$ are indistinguishable.*[3] *That is, for every polynomial-time distinguishing algorithm D, there is a negligible function*[4] negl*, so that*

$$\forall x \in X \quad \left| \Pr\left[D(x, f(x), F(x)) = 1\right] - \Pr\left[D\left(x, f(x), S(1^{|x|}, f(x))\right) = 1\right]\right| \le \mathsf{negl}(|x|)$$

where the probabilities are taken over the randomness of F, S and D. (This is a uniform *definition in the sense that D is required to be an algorithm. One can consider also a* non-uniform *version of the definition where D is only required to be a family of poly-size circuits. In such a case some of the inputs to D can be omitted; see Remark 1 at the end of Section 3.1.)*

The following definition of approximation applies to *minimization* problems. The definition for maximization problems is similar.

Definition 2 (Approximation). *Let f be a function and F be an algorithm, as above. The algorithm F is an* approximation within ratio $\alpha > 1$ *(a.k.a.* α-approximation*) for the function f if it runs in polynomial time and for all $x \in X$ we have $F(x) \ge f(x)$ (with probability 1), and $\mathbb{E}[F(x)] \le \alpha f(x)$.*

We say that f admits a polynomial time approximation scheme (PTAS) *if for every fixed $\epsilon > 0$ there exists a $(1+\epsilon)$-approximation for f. We say that f admits a* fully polynomial time approximation scheme (FPTAS) *if for every fixed $\epsilon > 0$ there exists for f a $(1+\epsilon)$-approximation whose running time is polynomial also in $1/\epsilon$.*

We say that F is a functionally private α-approximation *for f if it is functionally private w.r.t. f and it is an α-approximation for f.*

Definition 3 (Polynomially-bounded). *A function f is said to be* polynomially-bounded *if there exists a polynomial b such that $f(x) \le b(|x|)$ for all x.*

An algorithm F is polynomially-bounded *if there exists a polynomial b such that, for all x, $F(x) \le b(|x|)$ with probability 1.*

A simulator S is polynomially-bounded *if there exists a polynomial b such that, for all x and z, $S(1^{|x|}, z) \le b(|x|)$ with probability 1.*

For the problems that we are interested in, it is obvious that f is polynomially-bounded, and then we can assume w.l.o.g. that both the approximation algorithm F and the simulator S are also polynomially-bounded. Indeed, we use the assumption that F and S are polynomially bounded in Proposition 1 below.

Note that when f is an integral-valued, polynomially-bounded function, the notion of indistinguishability that we use coincides with statistical closeness.

2.1 Comparison with the definitions of Feigenbaum et al.

We discuss below several technical differences between our definitions and those of Feigenbaum et al. [7].

[1]The definitions below extend also to randomized functions f; however, the functions f that we deal with in this work are deterministic in nature – they are the solutions of optimization problems.

[2]It is sometimes convenient, and often unavoidable, to let F leak the *size* of x. The definition below indeed makes that exception.

[3]Note that although $f(x)$ is fully determined by x, it is important that $f(x)$ is given explicitly as part of the distributions, because in our context f is a computationally-hard function.

[4]A function is negligible if it tends to zero faster than any inverse polynomial.

Single-argument vs. multi-argument. The definition of Feigenbaum et al. [7] views f as a multi-argument function, since in the setting of multi-party computation, the input x is partitioned among the parties.[5] In their definition of *t-privacy*, the simulator has access to $f(x)$ as well as to t of f's arguments. Functional privacy is then defined as 0-privacy, i.e. the case where the simulator has access only to $f(x)$.

In Definition 1 above, we ignore the way the input x is distributed among the parties.[6] We argue that the issue of how the inputs are partitioned among players is orthogonal (in our context) to whether f has a private approximation:

- Whenever we prove an impossibility result for a private approximation of a function $f(x)$, it immediately follows from our Definition 1 that even in the very simple setting where one player holds the whole input x, other players have no input (except for $1^{|x|}$), and all players need to learn the output, a private approximation of f is impossible. More complicated partitions of the input x among the players are typically no easier, and their proof follows from our arguments with appropriate modifications.

- Whenever we present a private approximation for f, the corresponding algorithm F can be transformed into a secure multiparty protocol, by applying general-purpose transformations (such as [10]) to the circuit computing F (these tools already take care of the partition of inputs among players). Note that Feigenbaum et al. [6, 7] avoid using such general-purpose transformations (in their Hamming distance protocol) since they aim at having extremely efficient protocols. We deal with functions that in general are not believed to have polynomial-time exact evaluations and so the ability of doing anything privately is not clear a-priori.

Information theoretic vs. computational privacy. The definition of Feigenbaum et al. [7] requires the simulator to produce a distribution which is identical to $F(x)$. Private approximability results, such as those of [7], are clearly stronger if they are proved for such *information theoretic privacy*.

However, our relaxed notion of "indistinguishable" distributions (see Definition 1 above) is more appropriate for private inapproximability results. Our *computational privacy* requirement is weaker than that of [7], and thus all our impossibility results extend to their definition. In particular, impossibility of this weaker requirement shows that the inapproximability does not follow merely from the technical impossibility of creating identical distributions.

Approximation criteria. There is also a difference between our Definition 2 above and the notion of approximation used in [7]. They use the notion of (ϵ, δ)-approximation, that requires F to satisfy $(1-\epsilon)f(x) \le F(x) \le (1+\epsilon)f(x)$ that with probability at least $1-\delta$. Our definition is more commonly used in the literature dealing with approximation of NP-hard optimization problems [13, 2] (and is also simpler to work with in our context). We remark that the difference between the definitions does not change any of the results qualitatively and has only a minor quantitative impact.

[5]Indeed, the main motivation for the work of Feigenbaum et al. was massive data sets (such as those produced by network monitoring and operating systems), where the data is split among several entities.

[6]Although in our definition the simulator is given only $f(x)$, exactly as in the 0-privacy of Feigenbaum et al. [7].

3. PRIVATE APPROXIMATIONS OF NP-HARD PROBLEMS

3.1 Inapproximability results

The basic tool that we use to obtain inapproximability results is developed next. The intuition behind it is as follows: Let f be an objective function of some optimization problem, such that it is NP-hard to distinguish between $f(x) = z$ and $f(x) = z+1$ for some z and input length $|x| = n$. Suppose that f is privately approximated by an algorithm F, and assume for now that F is deterministic. Since F is private, we have that $F(x) = S(1^n, f(x))$ and thus all instances x with the same $f(x)$ have the same value $F(x)$. In particular, F has the same value $\tilde{F}(z)$ on all instances with $f(x) = z$, and the same value $\tilde{F}(z+1)$ on all the instances with $f(x) = z+1$. But F cannot distinguish between $f(x) = z$ and $f(x) = z+1$ instances (since it is NP-hard to do so). Hence, it must be the case that $\tilde{F}(z) = \tilde{F}(z+1)$.

Assume now that for every z within some range $[\mathsf{Lo}, \mathsf{Hi}]$, it is NP-hard to distinguish between instances x such that $f(x) = z$ and instances x such that $f(x) = z+1$. Then, by the argument above, it must be the case that $\tilde{F}(\mathsf{Lo}) = \tilde{F}(\mathsf{Lo}+1) = \cdots = \tilde{F}(\mathsf{Hi})$, and therefore, the approximation ratio of F cannot be smaller than Hi/Lo. This argument is made formal below. In particular, we need to extend it to the case where F is randomized.

The expected value of $F(x)$. The core idea in the informal argument above, is that when F is private, $F(x)$ does not really depend on x itself, but only on the value of $f(x)$. To handle a probabilistic F, we use the expected value of a distribution as its "representative": The role of $F(x)$ in the above argument is played by $\mathbb{E}[F(x)]$, and that of $\tilde{F}(z)$ is played by $\mathbb{E}[S(1^n, z)]$. The property that we need is that all inputs x with the same value $f(x)$ have (roughly) the same value $\mathbb{E}[F(x)]$, and this is (roughly) the value $\mathbb{E}[S(1^n, z)]$. The following proposition proves this property for polynomially-bounded case (technically, it suffices to assume that F is polynomially-bounded).

PROPOSITION 1. *Let F be functionally private w.r.t. f. If F is polynomially-bounded, then there exists a polynomially-bounded simulator S as in Definition 1, such that for all $x \in X$, $\left|\mathbb{E}[S(1^{|x|}, f(x))] - \mathbb{E}[F(x)]\right| = \mathsf{negl}(|x|)$, where $\mathsf{negl}(\cdot)$ is some negligible function.*

PROOF. Let $b(\cdot)$ be the polynomial bound on F, and let S be the simulator for F that is guaranteed by Definition 1. We can assume w.l.o.g. that S is also bounded by $b(\cdot)$, as otherwise we can modify S so that it never outputs anything larger than $b(n)$ on an input $(1^n, *)$, and the result will still be a good simulator.

Assume (toward contradiction) that for some polynomial $p(\cdot)$ and infinitely many x's,

$$\left|\mathbb{E}[S(1^{|x|}, f(x))] - \mathbb{E}[F(x)]\right| > 1/p(|x|)\ .$$

We will show a distinguisher D that has, on these x's, a large advantage in distinguishing between the distributions $F(x)^m$ and $S(1^{|x|}, f(x))^m$, where $m = 8|x|\cdot(p(|x|)\cdot b(|x|))^2$. Since m is polynomial in $|x|$ and since both $F(x)$ and $S(1^{|x|}, f(x))$ are efficiently sampleable, it will follow from a standard hybrid argument that we can also distinguish between $F(x)$ and $S(1^{|x|}, f(x))$, contradicting the privacy of F.

The distinguisher D is given as input $\langle x, z, y_1, \ldots, y_m\rangle$, and it needs to decide whether the y's were drawn from the distribution $S(1^{|x|}, z)$ or from $F(x)$. To do that, D estimates the expected value of both $S(1^{|x|}, z)$ and $F(x)$, using m samples of each. It outputs 0 if the average of the y_i's is closer to the estimated expected value of $S(1^{|x|}, z)$, and 1 otherwise. Using the Hoeffding bound, it can be seen that this distinguisher has advantage of at least $1 - 6e^{-|x|}$. $\square$

Remark 1. *The distinguishing algorithm D that is described above, uses x to compute m samples of $F(x)$, and uses $f(x)$ in order to compute m samples of $S(1^{|x|}, f(x))$. When considering a non-uniform distinguisher D, there is no need to provide it with x and $f(x)$; in this case a sequence of x's and the corresponding $f(x)$'s can be "wired" into the circuit (and need not be given explicitly).*

3.1.1 Sliding-window reductions

Let f be an objective function of some (NP-hard) minimization problem O. We say that O has a *sliding-window reduction* from some NP-hard language L, if there exists a polynomial time reduction $L \le O$ with the following structure:[7]

- The reduction takes as input an instance v of L and an integer z, and produces as output an instance x of O. Moreover, there exists an increasing, polynomially bounded function $\mathsf{Ex}(\cdot)$ (for Expansion), such that on input instance v of size n (and any integer z), the size of the output instance x is exactly $N = \mathsf{Ex}(n)$.
- There exist functions $\mathsf{Lo}(n), \mathsf{Hi}(n)$ that are polynomially-bounded and can be computed in time polynomial in n, such that on input instance v of size n and integer $z \in \{\mathsf{Lo}(n), \ldots, \mathsf{Hi}(n)\}$, the reduction returns an instance x of O such that
  - if $v \in L$ then $f(x) = z$
  - if $v \notin L$ then $f(x) \ge z + 1$

Essentially, the above requirement says that between the bounds Lo and Hi, it is NP-hard to distinguish between instances of O with value z and $z+1$, and moreover, there is a single, parameterized, reduction that shows all these NP-hardness results. We show that in this case, there is no private approximation for O that can achieve ratio better than Hi/Lo.

Lemma 2. *Assume that f is an objective function of a minimization problem, that has a sliding-window reduction from some NP-hard language L, with functions $\mathsf{Lo}(\cdot), \mathsf{Hi}(\cdot)$ and $\mathsf{Ex}(\cdot)$, as above. Then, unless $NP \subseteq BPP$, f does not have a functionally private approximation within ratio*

$$\alpha(N) = \left(1 - \frac{1}{\mathrm{poly}(N)}\right) \cdot \frac{\mathsf{Hi}(\mathsf{Ex}^{-1}(N))}{\mathsf{Lo}(\mathsf{Ex}^{-1}(N))}$$

where $\mathsf{Ex}^{-1}(\cdot)$ is the "inverse" of $\mathsf{Ex}(\cdot)$, namely, $\mathsf{Ex}^{-1}(N) = \max\{n : \mathsf{Ex}(n) \le N\}$.

Proof. Assume, to the contrary, that there exists an algorithm F that is a functionally private approximation for f within ratio $\alpha(N) = (1-1/q(N))\cdot\mathsf{Hi}(\mathsf{Ex}^{-1}(N))\ /\ \mathsf{Lo}(\mathsf{Ex}^{-1}(N))$ for some polynomial q. We show how to use F to solve the NP-hard language L in probabilistic polynomial time.

Intuition. On a high level, the argument proceeds as follows: Fix some input length n, and denote $N = \mathsf{Ex}(n)$, $\mathsf{Lo} = \mathsf{Lo}(n)$, and $\mathsf{Hi} = \mathsf{Hi}(n)$. Recall that we assume that for all $z \in [\mathsf{Lo}, \mathsf{Hi}]$ it is NP-hard to distinguish instances with $f(x) = z$ from instances with $f(x) \ge z + 1$. Consider the output of the simulator S on the inputs $(1^N, z)$ for $z \ge \mathsf{Lo}$. (The output on $(1^N, z)$ is essentially the output distribution of F on any input x of size N with value $f(x) = z$.) Since F is an α-approximation, we have

$$\begin{aligned}
\mathbb{E}\left[S(1^N, \mathsf{Lo})\right] &\le \alpha(N)\cdot\mathsf{Lo}\\
&= \left(\left(1 - \frac{1}{\mathrm{poly}(N)}\right)\cdot\frac{\mathsf{Hi}}{\mathsf{Lo}}\right)\cdot\mathsf{Lo}\\
&= \left(1 - \frac{1}{\mathrm{poly}(N)}\right)\cdot\mathsf{Hi}\\
&\le \left(1 - \frac{1}{\mathrm{poly}(N)}\right)\cdot\mathbb{E}\left[S(1^N, \mathsf{Hi})\right]
\end{aligned}$$

So there must be an index $z_0 \in \{\mathsf{Lo}, \ldots, \mathsf{Hi}\}$ such that $\mathbb{E}[S(1^N, z_0)]$ is noticeably smaller than $\mathbb{E}[S(1^N, z)]$ for any $z > z_0$. To decide if $v \in L$, we apply to it the sliding-window reduction with parameter z_0, and get an instance x for O. We then estimate $\mathbb{E}[F(x)]$ and compare it to $\mathbb{E}[S(1^N, z_0)]$. If they are close enough we accept v, and otherwise we reject it. A more formal description follows.

Technicalities. We first need to make sure that the approximation algorithm F is polynomially-bounded, so that we can use Proposition 1. If it is not, we can modify the problem O and the algorithm F as follows: Let $b(\cdot)$ be a polynomial upper-bound on $\mathsf{Hi}(\cdot)$. The modified problem O' has an objective function $f'(x) = \min\{f(x), b(|x|)\}$, and the modified approximation algorithm is $F'(x) = \min\{F(x), b(|x|)\}$. It is easy to see that F' is a polynomially-bounded private approximation for f', and that the reduction $L < O$ is also a sliding-window reduction $L < O'$ with the same parameters. From now on, we assume w.l.o.g. that the original algorithm F is already polynomially-bounded, and let S be the polynomially-bounded simulator from Proposition 1.

The algorithm. We now show an algorithm A for deciding the language L. On input instance v for L, A sets $n := |v|$, $N := \mathsf{Ex}(n)$, $\mathsf{Lo} := \mathsf{Lo}(n)$, $\mathsf{Hi} := \mathsf{Hi}(n)$, and $\delta := 1/(q(N)\cdot\mathsf{Hi})$. It then estimates $\mathbb{E}[S(1^N, z)]$ for $z = \mathsf{Lo}, \ldots, \mathsf{Hi}$, each with (additive) accuracy of $\delta/8$ and exponentially small error probability. (This is done in time polynomial in n, since S is polynomially-bounded and so are all the functions $\mathsf{Ex}, \mathsf{Lo}, \mathsf{Hi}$

[7] More generally, we can have a reduction from any NP-hard decision problem, e.g. a promise problem.

and q.) Denote A's estimate for $\mathbb{E}[S(1^N, z)]$ by $\bar{\mu}_z$. A sets

$$u_z := \begin{cases} z & \text{if } \bar{\mu}_z < z \\ \bar{\mu}_z & \text{if } \bar{\mu}_z \in [z,\ \alpha z] \\ \alpha z & \text{if } \bar{\mu}_z > \alpha z \end{cases}$$

Then, A lets z_0 be any index such that $u_{z_0} < u_z - \delta$ for all $z > z_0$, and also $u_{z_0} < \mathsf{Hi} - \delta$. (Below we prove that such an index z_0 must always exist.) Next, A applies the sliding-window reduction to (v, z_0), and gets an instance x of O, so that $f(x) = z_0$ if $v \in L$ and $f(x) \geq z_0 + 1$ if $v \notin L$. It then estimates $\mathbb{E}[F(x)]$ with accuracy $\delta/4$ and exponentially small error probability. Denoting this estimate by $\bar{\lambda}$, the input v is accepted if $\bar{\lambda} < u_{z_0} + \delta/2$, and it is rejected otherwise.

Analysis of the algorithm. With probability at least $2/3$, all the estimates that A makes (i.e., the $\bar{\mu}_z$'s and $\bar{\lambda}$) are within the specified accuracy range. Below we assume that this is indeed the case, and show that in this case A will accept v if and only if $v \in L$.

First, we prove that there must always exist an index z_0 as above. By construction, we have $u_z \in [z, \alpha z]$ for all z, and therefore

$$\begin{aligned} u_{\mathsf{Lo}} &\leq \alpha \cdot \mathsf{Lo} \\ &\leq \left(\left(1 - \frac{1}{q(N)}\right) \cdot \frac{\mathsf{Hi}}{\mathsf{Lo}}\right) \cdot \mathsf{Lo} \\ &= \mathsf{Hi} \cdot \left(1 - \frac{1}{q(N)}\right) \\ &< \mathsf{Hi} - \frac{1}{q(N)} \;\leq\; u_{\mathsf{Hi}} - \frac{1}{q(N)} \end{aligned}$$

As there are only $\mathsf{Hi} - \mathsf{Lo}$ values of z between Lo and Hi, there must exist an index z_0 as above.

Next, assume that $v \in L$. Then the sliding-window reduction produces an instance x such that $f(x) = z_0$. Proposition 1 implies that $|\mathbb{E}[S(1^N, z_0)] - \mathbb{E}[F(x)]| = \mathsf{negl}(N)$, and since $|\bar{\mu}_z - \mathbb{E}[S(1^N, z)]| < \delta/8$, we have $|\bar{\mu}_{z_0} - \mathbb{E}[F(x)]| < \delta/8 + \mathsf{negl}(N) < \delta/4$. The only case where $u_{z_0} \neq \bar{\mu}_{z_0}$, is if the latter is not in the range $[z_0, \alpha z_0]$, in which case u_{z_0} is set to the closest point to $\bar{\mu}_{z_0}$ in this range. But the properties of the approximation algorithm F tells us that $\mathbb{E}[F(x)] \in [z_0, \alpha z_0]$, so we conclude that also $|u_{z_0} - \mathbb{E}[F(x_z)]| < \delta/4$. By the accuracy of A's estimate of $\mathbb{E}[F(x_z)]$, we have $|\bar{\lambda} - \mathbb{E}[F(x_{z_0})]| < \delta/4$, which implies $|u_{z_0} - \bar{\lambda}| < \delta/2$, so A will accept its input.

Finally, assume that $v \notin L$. This means that the instance x that A gets from the reduction has $f(x) \geq z_0 + 1$. Here there are two subcases: either $f(x) \leq \mathsf{Hi}$, or $f(x) > \mathsf{Hi}$. In the first case, we have $f(x) = z'$ for some $z' \in \{z_0 + 1, \ldots, \mathsf{Hi}\}$. By the same arguments as above, we would get $|u_{z'} - \bar{\lambda}| < \delta/2$. However, since $z' > z_0$, then we have $u_{z'} > u_{z_0} + \delta$, and therefore $\bar{\lambda} > u_{z_0} + \delta/2$, so A will reject its input. The second subcase is proved similarly, this time using the fact that $f(x) > \mathsf{Hi}$ implies that also $\mathbb{E}[F(x)] > \mathsf{Hi} > u_{z_0} + \delta$. □

Extensions. Lemma 2 can be extended in several ways:

Exponential gap. For a deterministic approximation algorithm F, the proof of Lemma 2 can be extended to $\mathsf{Hi}(n)$ that is exponential in n, as follows.

- If F is monotone (i.e., $F(x) \leq F(x')$ whenever $f(x) \leq f(x')$): Since F is deterministic, it suffices to find the largest index $z_0 \geq \mathsf{Lo}$ for which $u_{z_0} < \mathsf{Hi}$, and since F is monotone, this can be done efficiently using binary search.
- If algorithm A is allowed to be non-uniform, and then the inapproximability of f holds unless NP $\subseteq$ P/poly: Since F is deterministic, it suffices to find the largest index $z_0 \geq \mathsf{Lo}$ for which $u_{z_0} < \mathsf{Hi}$, and since F is non-uniform, and this z_0 can be obtained as an advice (as it depends on $n = |v|$ but not on v).
- If there is equality on both sides of the sliding-windows reduction (namely, if $v \notin L$ then $f(x) = z+1$): In this case it suffices to find an index $z_0 \in [\mathsf{Lo}, \mathsf{Hi} - 1]$ with $u_{z_0} \neq u_{z_0+1}$, and since $u_{\mathsf{Lo}} \neq u_{\mathsf{Hi}}$, this can be done efficiently using binary search.

Inequality on both sides. Lemma 2 can be extended, as follows, to the case where we only know that $f(x) \leq z$ when $v \in L$ and $f(x) \geq z + 1$ when $v \notin L$. (Recall that the definition of sliding-window reduction above had $f(x) = z$ when $v \in L$.) The additional property that we need is that for every $v \in L$, there is a $\Delta_v \in [0, \mathsf{Lo}]$ so that for every $z \in [\mathsf{Lo}, \mathsf{Hi}]$, applying the reduction to (v, z) returns an instance x with $f(x) = z - \Delta_v$. We stress that Δ_v does not depend on z, which is usually not much of a limitation, since such reductions are achieved by starting from a standard reduction, and then using padding techniques to "shift" the value of $f(x)$ by some known quantity, in order to "hit" the right z. In such a reduction, the Δ_v follows from the standard reduction, and thus we have the same Δ_v for all shift amounts.

To decide if $v \in L$, the algorithm tries to "guess" the value of Δ_v (assuming that $v \in L$). It first computes the u_z's and finds the index z_0, as in the proof of Lemma 2. For each possible value, $\Delta = 0, 1, \ldots, \mathsf{Lo}$, the algorithm applies the sliding-window reduction on $(v, z_0 + \Delta)$ to get an instance x_Δ, and estimates $\mathbb{E}[F(x_\Delta)]$ with accuracy $\delta/4$ and exponentially small error probability. Call this estimate $\bar{\lambda}_\Delta$. If $v \in L$, then the same arguments as in the proof of Lemma 2 imply that for at least one value of Δ we have $|\bar{\lambda}_\Delta - u_{z_0}| < \delta/2$ (namely, when we guess the right value, $\Delta = \Delta_v$). If $v \notin L$, then for any $\Delta \in [0, \mathsf{Lo}]$ we have $f(x_\Delta) \geq z_0 + \Delta + 1 > z_0$, and therefore $\bar{\lambda}_\Delta > u_{z_0} + \delta/2$.

3.1.2 The minimum vertex cover problem

We demonstrate how Lemma 2 can be used to obtain an inapproximability result for the minimum vertex cover problem. Let $G = (V, E)$ be a graph, and define $f_{VC}(G)$ as the minimum size of a vertex cover[8] of G.

Theorem 3. *For every fixed $\varepsilon > 0$, the function $f_{VC}(G)$ has no functionally private (probabilistic polynomial time) approximation within ratio $N^{1-\varepsilon}$, where N is the number of vertices in G, unless NP $\subseteq$ BPP.*

Proof. We start with a known reduction from SAT to VC (say, the reduction due to Feige et al. [5]). On input formula ϕ, the reduction produces a graph H on n vertices, and an integer $s \leq n$, such that if ϕ is satisfiable then $f_{VC}(H) = s$ and if ϕ is not satisfiable then $f_{VC}(H) \geq s + 1$.

[8] A vertex cover of a graph is a subset of the vertices that contains at least one endpoint of every edge.

In these reductions, n and s depend only on the size of ϕ, not on ϕ itself. Furthermore, they can be computed efficiently from the size of ϕ.

Our sliding-window reduction, given the graph H from above and an integer z (within a range that is described below), and constructs another graph G on $N = \lceil n^{1+1/\varepsilon} \rceil$ vertices, as follows: G consists of all the vertices and edges of H, and also of a clique on $i = z - s + 1$ additional vertices, and an independent set on $\lceil n^{1+1/\varepsilon} \rceil - n - i$ additional vertices. We note the following properties of G:

- The number of vertices in G is always $N = \lceil n^{1+1/\varepsilon} \rceil$. In the language of Lemma 2, we have $N = \mathsf{Ex}(n) = \lceil n^{1+1/\varepsilon} \rceil$ (and therefore $n = \mathsf{Ex}^{-1}(N) = \lfloor N^{\varepsilon/(1+\varepsilon)} \rfloor$).
- The size of the smallest vertex cover in G is always $f_{VC}(G) = f_{VC}(H) + i - 1$. This means that if ϕ is satisfiable, then $f_{VC}(G) = s + i - 1 = z$, and otherwise $f_{VC}(G) \geq z + 1$.

Since we can have i between 1 and $\lceil n^{1+1/\varepsilon} \rceil - n$, it follows that for this reduction we have $\mathsf{Lo}(n) = s$ and $\mathsf{Hi}(n) = s + \lceil n^{1+1/\varepsilon} \rceil - n$. Hence, we have

$$\frac{\mathsf{Hi}(n)}{\mathsf{Lo}(n)} = \frac{s + \lceil n^{1+1/\varepsilon} \rceil - n}{s} \geq \frac{n^{1+1/\varepsilon}}{n} = n^{1/\varepsilon}$$

where the inequality holds since $s \leq n$. Applying Lemma 2, we conclude that $f_{VC}(G)$ cannot be privately approximated within a ratio of

$$\alpha(N) = \left(1 - \frac{1}{\mathrm{poly}(N)}\right) \cdot \frac{\mathsf{Hi}(\mathsf{Ex}^{-1}(N))}{\mathsf{Lo}(\mathsf{Ex}^{-1}(N))}$$

and using

$$\begin{aligned} \frac{\mathsf{Hi}(\mathsf{Ex}^{-1}(N))}{\mathsf{Lo}(\mathsf{Ex}^{-1}(N))} &\geq \left(\mathsf{Ex}^{-1}(N)\right)^{1/\varepsilon} \\ &= \left(\lfloor N^{\frac{\varepsilon}{1+\varepsilon}} \rfloor\right)^{1/\varepsilon} \approx N^{1/(1+\varepsilon)} > N^{1-\varepsilon} \end{aligned}$$

the proof is complete. $\square$

REMARK 2. *It should be noted that although the reduction in the proof above starts from SAT, the sliding-window reduction is actually from a promise problem (of vertex cover), where we are given a graph G with $f_{VC}(G) \geq s$, and we need to decide whether $f_{VC}(G) = s$ or $f_{VC}(G) > s$.*

3.1.3 Other examples

The requirements of Lemma 2 are satisfied by many other NP-hard functions. We list (without proofs) a few other functions for which this lemma applies.

MAX-SAT: The MAX-SAT problem can be trivially approximated within ratio 0.5, since for every CNF formula with N clauses, one can efficiently find an assignment that satisfies at least $N/2$ clauses.

If we do not require privacy, this ratio can be improved to 0.7846, as shown by Asano and Williamson [1]. However, if we do require privacy, one can apply Lemma 2 (or rather its maximization variant) to show that MAX-SAT cannot be privately approximated within ratio $0.5 + 1/N^{1-\varepsilon}$, for any fixed $\varepsilon > 0$, unless NP $\subseteq$ BPP.

Vertex cover in planar graphs: This NP-hard problem admits a PTAS (see [3]). However, a proof similar to that of Theorem 3, it can be shown that there is no private $N^{1-\epsilon}$ approximation for vertex cover in planar graphs unless NP $\subseteq$ BPP.

3.2 Private FPTAS for an NP-hard function

Although Lemma 2 can be used to prove inapproximability results for many NP-hard functions, such inapproximability results do not hold for every NP-hard optimization problem. Specifically, we can prove that:

THEOREM 4. *There exist NP-hard functions that admit a (deterministic) private FPTAS.*

PROOF. Consider the function $f(x) \stackrel{\text{def}}{=} 2^{|x|} + f_{VC}(x)$. Clearly, computing f is NP-hard (since f_{VC} is NP-hard). However, an algorithm F with $F(x) = 2^{|x|} + |x|$ is a functionally private α-approximation for f, with $\alpha(N) \leq \frac{2^N + N}{2^N} = 1 + \frac{N}{2^N}$. $\square$

4. ALMOST PRIVATE APPROXIMATIONS OF NP-HARD PROBLEMS

The following definition relaxes the privacy notion of Definition 1, allowing F to leak a limited amount of information about x. For similar notions, see [4, 11].

DEFINITION 4. *Let f be a function and F be an algorithm. The algorithm F* leaks at most k bits *of information w.r.t. f if there exists a polynomial time "simulator" S and a deterministic "hint" function $h : X \to \{1, \ldots, 2^k\}$, such that the distribution $\langle x, f(x), S(1^{|x|}, f(x), h(x)) \rangle$ is computationally indistinguishable from $\langle x, f(x), F(x) \rangle$.*

Note that k may be non-integral as long as 2^k is integral. (Also note that we do not require that h be efficiently computable).

PROPOSITION 5. *F is functionally private w.r.t. f if and only if it leaks 0 bits of information w.r.t. f.*

REMARK 3. *We stress that the definition above is strong in the sense that even polynomially many samples of the distribution $F(x)$ do not leak more than k bits about the input.*

For motivation, consider $F(G)$ the 2-approximation for $f_{VC}(G)$ that is twice the size of a maximum matching in G (see e.g. [8, pp. 133-134]). Clearly, F is not private (as it leaks the size of the maximum matching in G). In an attempt to leak less information, suppose that we add some "random noise" to F. For example, define a modified scheme F', that first computes $F(G)$ and then outputs a random value in the range (say) $\{F(G) \ldots 2F(G)\}$. Intuitively, it may appear that $F'(G)$ provides a randomized 4-approximation that leaks much less information about G. However, although a single run *of $F'(G)$ may leak very little information,* polynomially many runs *of $F'(G)$ would almost surely leak $F(G)$ (by taking the minimum), and hence leak the maximum matching size.*

4.1 Upper bounds

We show that in contrast to Theorem 3, there exists a feasible constant approximation function for the vertex cover function that leaks only a single bit.

Claim 1. *There exists a 4-approximation for vertex cover that leaks one bit of information.*

Proof. Consider a (deterministic) 2-approximation for $f_{VC}(G)$, say the approximation that equals twice the size of a maximum matching in G. This function is not a private approximation since it reveals the size of the maximum matching in G, a value that cannot be computed given only $f_{VC}(G)$. Note that the maximum matching may take values between $f_{VC}(G)/2$, and $f_{VC}(G)$ and thus our 2-approximation leaks many bits. In the following we transform the 2-approximation into a function that leaks only a single bit of information, at the cost of a increasing the approximation ratio. Details follow.

Let $f'(G)$ be the 2-approximation for $f_{VC}(G)$ from above. Define $F(G)$ to be the value of $f'(G)$ rounded upwards to the closest integral power of 2:

$$F(G) = \min\{2^p : p \in \mathbb{N} \cup \{0\}, 2^p \geq f'(G)\}.$$

F is a 4-approximation for $f_{VC}(G)$ since $F(G) \geq f'(G) \geq f_{VC}(G)$ and $F(G) < 2f'(G) \leq 4f_{VC}(G)$.

To show that F leaks only one bit we note that, for every z, if we look at all the graphs G satisfying $f_{VC}(G) = z$ then there are only two possible values for $F(G)$. The hint function $h(G)$ may thus be used select one of these two possible values. More precisely, consider the power of 2 between $f_{VC}(G)$ and $2f_{VC}(G)$, i.e. $f_{VC}(G) \leq 2^p < 2f_{VC}(G)$, and let

$$h(G) = \begin{cases} 1 & \text{if } f'(G) \leq 2^p \\ 2 & \text{if } f'(G) > 2^p \end{cases}$$

The simulator is given $f_{VC}(G)$ and $h(G)$. If $h(G) = 1$ the simulator outputs 2^p (trivial computation given $f_{VC}(G)$), otherwise it outputs 2^{p+1}. It is easy to verify that the simulator always outputs $F(G)$ and hence F is a 4-approximation that leaks one bit of information. $\square$

Similar arguments hold in the general case, as stated in the following theorem.

Theorem 6. *If f has a deterministic α-approximation, then it has an $\alpha^{1+1/(2^k-1)}$-approximation that leaks at most k bits of information.*

Proof. Let $f'(x)$ be a polynomial time computable deterministic α-approximation for $f(x)$. Let $F(x)$ be the value of $f'(x)$ rounded upwards to the closest power of $\beta = \alpha^{1/(2^k-1)}$. I.e.

$$F(x) = \min\{\beta^p : p \in \mathbb{N} \cup \{0\}, \beta^p \geq f'(x)\}$$

It follows that F is a (polynomial time) $\alpha \cdot \beta$ approximation, since $F(x) \geq f'(x) \geq f(x)$ and $F(x) < \beta f'(x) \leq \alpha\beta f(x)$.

To show that F leaks at most k bits of information, we exhibit a simulator S and hint function h in accordance with Definition 4.

Observe that $F(x)$ is a power of β that satisfies $f(x) \leq F(x) < \alpha\beta f(x)$. Thus, given $f(x)$ there can be at most $\lceil \log_\beta \alpha\beta \rceil = 1 + \lceil \log_\beta \alpha \rceil = 2^k$ possible values for $F(x)$, and all of them are computable from $f(x)$ in polynomial time. The additional k bits of information given by the hint function $h(x)$ are used to select one of these 2^k values. Formally, let $h(x)$ be the value j for which $\beta^{j-1} \leq \frac{F(x)}{f(x)} < \beta^j$. (Note that $\beta^{2^k} = \alpha\beta$ so there exists such a j with $1 \leq j \leq 2^k$).

The simulator S is given $f(x)$ and $h(x)$, and outputs the value p which is an integral power of β that and satisfies $\beta^{h(x)-1} f(x) \leq p < \beta^{h(x)} f(x)$. It is easy to verify that $S(1^{|x|}, f(x), h(x)) = F(x)$. Hence, F is an $\alpha\beta = \alpha^{1+1/(2^k-1)}$ approximation that leaks at most k bits. $\square$

Remark 4. *The simulators presented in Claim 1 and Theorem 6 are stronger than required in Definition 4 since they produce distribution identical to the approximation function F. See the discussion in Section 2.1.*

4.2 Lower bounds

In this section we give some evidence that the upper bound of Theorem 6 may be tight. Specifically, we show that for a certain type of problems, a (standard) inapproximability result within a ratio of α, implies inapproximability within a ratio of $(\alpha/(1+\varepsilon))^2$ (for any fixed $\varepsilon > 0$), with respect to deterministic approximations that leak only one bit.

For the rest of this section, fix some $\varepsilon > 0, \alpha > 1+\varepsilon$ and denote $\tilde{\alpha} = \alpha/(1+\varepsilon)$. Let f be the objective function of some optimization problem O, and let F be a *deterministic* $\tilde{\alpha}^2$-approximation for f that leaks only one bit.

4.2.1 Tunable sliding-window reductions

We now present a sketch of our argument, which uses a certain type of sliding-window reduction for the problem O, in order to show that the assumption that F is a $\tilde{\alpha}^2$-approximation leads to contradiction (namely, P = NP). The idea here is somewhat similar to (but more complicated than) Lemma 2.

Intuition. As before, we begin by observing that the requirement of leaking just one bit implies some structural constraints on F. Specifically, for every input size n and every value z, there are at most two different answers that F can give on instances x of size $|x| = N$ with $f(x) = z$.[9] We denote these values by $\tilde{F}^{(1)}(z), \tilde{F}^{(2)}(z)$, and assume w.l.o.g. that $\tilde{F}^{(1)}(z) \leq \tilde{F}^{(2)}(z)$. We denote by $\tilde{F}(z)$ the set $\{\tilde{F}^{(1)}(z), \tilde{F}^{(2)}(z)\}$. Since F is an $\tilde{\alpha}^2$ approximation, we get that for all z,

$$z \leq \tilde{F}^{(1)}(z) \leq \tilde{F}^{(2)}(z) \leq \tilde{\alpha}^2 z \tag{1}$$

Let L be an NP-hard language, and assume that there is a (variant of) sliding-window reduction from L to O, showing that it is NP-hard to distinguish between $f(x) = z$ and $f(x) = \alpha z$ for all z within a sufficiently large range $[\mathsf{Lo}, \mathsf{Hi}]$ (say $\mathsf{Hi}/\mathsf{Lo} > \alpha^5$). For the argument below, we require equality in both cases. I.e., we assume that for $v \in L$ the reduction gives us $f(x) = z$, and for $v \notin L$ it gives us $f(x) = \alpha z$.

Let $z \in [\mathsf{Hi}, \mathsf{Lo}]$ be some value. Since F cannot distinguish between $f(x) = z$ and $f(x) = \alpha z$, it follows that $\tilde{F}(z)$ and $\tilde{F}(\alpha z)$ must have at least one value in common.

[9]Namely, the values $S(1^N, z, 1)$ and $S(1^N, z, 2)$ that the simulator outputs on input z and hint values 1 and 2.

Specifically, it can be shown (using the fact that F is an $\tilde{\alpha}^2$-approximation, with $\tilde{\alpha} < \alpha$), that $\tilde{F}^{(2)}(z) = \tilde{F}^{(1)}(\alpha z)$.

Assume further, that the sliding-window reduction can be "tuned down", so that it creates any desired gap β with $\tilde{\alpha} < \beta < \alpha$. Namely, we still have the same parameters $\mathsf{Ex}, \mathsf{Lo}, \mathsf{Hi}$, but now for $v \notin L$ we get $f(x) = \beta z$ (rather than $f(x) = \alpha z$). It is usually quite easy to weaken the original reduction to obtain such a smaller gap.

Applying the same arguments as above, we can show that here too we must have $\tilde{F}^{(2)}(z) = \tilde{F}^{(1)}(\beta z)$. Taking these two arguments together, we conclude that $\tilde{F}^{(1)}(\beta z) = \tilde{F}^{(1)}(\alpha z)$. In other words, for any value w (which we think of as $w = \beta z$), we have $\tilde{F}^{(1)}(w) = \tilde{F}^{(1)}(w \cdot (\alpha/\beta))$. Hence we get (roughly)

$$\begin{array}{rcl} \tilde{F}^{(1)}(\mathsf{Lo}) & = & \tilde{F}^{(1)}(\mathsf{Lo} \cdot (\alpha/\beta)) \;\; = \\ \cdots & = & \tilde{F}^{(1)}(\mathsf{Lo} \cdot (\alpha/\beta)^r) \;\; = \;\; \tilde{F}^{(1)}(\mathsf{Hi}) \end{array}$$

But this is a contradiction, since $\tilde{F}^{(1)}(\mathsf{Lo}) \leq \tilde{\alpha}^2\mathsf{Lo}$ and $\tilde{F}^{(1)}(\mathsf{Hi}) \geq \mathsf{Hi}$, and the values Lo and Hi are too far apart.

FORMAL ARGUMENT. We introduce the notion of a *tunable sliding-window reduction*, with parameters $\mathsf{Ex}, \mathsf{Lo}, \mathsf{Hi}$ and α, where $\mathsf{Lo}(n)$ is not bounded by a constant, $\mathsf{Hi}(n)$ is polynomially-bounded, and they both can be computed in time that is polynomial in n. Specifically, this is a polynomial time reduction $L \leq O$ with the following structure: The reduction takes as input an instance v of L (with $|v| = n$), an integer $z \in \{\mathsf{Lo}(n), \ldots, \mathsf{Hi}(n)\}$, and a real number $\beta \leq \alpha$, and produces as output an instance x of size $N = \mathsf{Ex}(n)$ of the problem O, such that

- if $v \in L$ then $f(x) = z$,
- if $v \notin L$ then $f(x) = \lfloor \beta z \rfloor$.

That is, we have one, parameterized, reduction to show that within the range $[\mathsf{Lo}, \mathsf{Hi}]$ it is NP-hard to distinguish $f(x) = z$ from $\lfloor \beta z \rfloor$ for every $\beta < \alpha$.

Let $\tilde{F}_N^{(1)}(z), \tilde{F}_N^{(2)}(z)$ denote the two values $S(1^N, z, 1)$ and $S(1^N, z, 2)$, assuming w.l.o.g. that $\tilde{F}^{(1)}(z) \leq \tilde{F}^{(2)}(z)$, and let $\tilde{F}_N(z) \stackrel{\text{def}}{=} \{\tilde{F}_N^{(1)}(z), \tilde{F}_N^{(2)}(z)\}$. We next prove that if $z' < z'' \leq \alpha z'$, then the two sets $\tilde{F}_N(z'), \tilde{F}_N(z'')$ cannot be disjoint (unless P = NP). Below we omit the subscript N when it is clear from the context.

CLAIM 2. *Assume that there is a tunable sliding-window reduction $L \leq O$ as above. Then there exists a polynomial time algorithm A, such that for every input length n, the following holds:*

If there exist $z', z'' \in [\mathsf{Lo}(n), \mathsf{Hi}(n)]$ such that $z' < z'' \leq \alpha z'$ and yet the sets $\tilde{F}(z'), \tilde{F}(z'')$ are disjoint, then for any input v of length n, $A(v) = 1$ if and only if $v \in L$.

PROOF. On input v of length n, algorithm A computes explicitly the sets $\tilde{F}(z) = \{S(1^N, z, 1), S(1^N, z, 2)\}$ for all $z \in [\mathsf{Lo}(n), \mathsf{Hi}(n)]$ by running the simulator S, and finds values z', z'' as above. Then, A performs the reduction on its input v, with parameters z' and $\beta = z''/z'$. The reduction returns an instance x of length N, for the problem O. Finally, A computes $F(x)$ and accepts v if and only if $F(x) \in \tilde{F}(z')$.

If $v \in L$, then $f(x) = z'$, and therefore $F(x) \in \tilde{F}(z')$ and A will accept. If $v \notin L$, then $f(x) = z''$ so $F(x) \in \tilde{F}(z'')$, and since the sets $\tilde{F}(z')$ and $\tilde{F}(z'')$ are disjoint, A will reject. $\square$

Claim 2 implies that unless P = NP, there are infinitely many n's such that for all z', z'' as above, the sets $\tilde{F}_N(z'), \tilde{F}_N(z'')$ are not disjoint. This is the property that we need to derive contradiction to F being an $\tilde{\alpha}^2$-approximation of f.

LEMMA 7. *Let f be an objective function of an NP-hard optimization problem. If f has a tunable sliding-window reduction, with parameters $\mathsf{Ex}, \mathsf{Lo}, \mathsf{Hi}$ and α, where $\mathsf{Hi}/\mathsf{Lo} > \alpha^5$, then for every fixed $\varepsilon > 0$, f does not have a deterministic $(\alpha/(1+\varepsilon))^2$ approximation that leaks at most one bit of information, unless $P = NP$.*

PROOF. Recall that we denote $\tilde{\alpha} = \alpha/(1+\varepsilon)$. Unless P = NP, there are infinitely many input lengths n for which the algorithm from Claim 2 fails, so for the entire proof we fix one such n that is sufficiently large. Denote $\mathsf{Lo} = \mathsf{Lo}(n)$, $\mathsf{Hi} = \mathsf{Hi}(n)$ and $N = \mathsf{Ex}(n)$.

Let $z_0 < z_1 < \ldots < z_m$ be (integer) values in the range $[\mathsf{Lo}, \mathsf{Hi}]$, such that $\tilde{\alpha} z_{i-1} < z_i \leq \alpha z_{i-1}$ (for $i = 1, \ldots, m$). By Claim 2, $\tilde{F}(z_i)$ is not disjoint from $\tilde{F}(z_{i-1})$. However, since F is an $\tilde{\alpha}^2$-approximation, we get from (1) that (for $i = 1, \ldots, m-1$) the sets $\tilde{F}(z_{i-1})$ and $\tilde{F}(z_{i+1})$ are disjoint, namely,

$$\begin{array}{ccccccc} \tilde{F}^{(1)}(z_{i-1}) & \leq & \tilde{F}^{(2)}(z_{i-1}) & \leq & \tilde{\alpha}^2 z_{i-1} & & \\ & < & z_{i+1} & \leq & \tilde{F}^{(1)}(z_{i+1}) & \leq & \tilde{F}^{(2)}(z_{i+1}) \end{array} \tag{2}$$

Therefore, $\tilde{F}(z_i)$ must have one value that intersects $\tilde{F}(z_{i-1})$ (and therefore at most $\tilde{\alpha}^2 z_{i-1}$), and one value that intersects $\tilde{F}(z_{i+1})$ (and therefore at least z_{i+1}), for $i = 1, \ldots, m-1$. We thus have

$$\begin{array}{lcl} \tilde{F}^{(1)}(z_i) \in \tilde{F}(z_{i-1}) & \text{and therefore} & \tilde{F}^{(1)}(z_i) \notin \tilde{F}(z_{i+1}) \\ \tilde{F}^{(2)}(z_i) \in \tilde{F}(z_{i+1}) & \text{and therefore} & \tilde{F}^{(2)}(z_i) \notin \tilde{F}(z_{i-1}) \end{array}$$

Hence, we must have for $i = 2, \ldots, m-1$

$$\tilde{F}^{(2)}(z_{i-1}) = \tilde{F}^{(1)}(z_i) \tag{3}$$

This structure is depicted in Figure 1.

Let k be a large enough integer so that $\beta \stackrel{\text{def}}{=} \alpha^{(k-1)/k} > \tilde{\alpha}$. Let $w_0, w_1, \ldots, w_r$ be the values $w_j \stackrel{\text{def}}{=} \lceil \mathsf{Lo} \cdot (\alpha^{j/k}) \rceil$, where w_r is the first value such that $w_r \geq \mathsf{Hi}$. For any $j \in \{2k, \ldots, r-k\}$, we can apply the argument from above to the sequence $z_0 = w_{j-2k}$, $z_1 = w_{j-k}$, $z_2 = w_j$, $z_3 = w_{j+k}$ (note that $z_i/z_{i-1} \approx \alpha$, and the rounding error can be "neglected" since $\mathsf{Lo} \gg 1/\varepsilon$), and therefore, by Eq. (3) we have $\tilde{F}^{(1)}(w_j) = \tilde{F}^{(2)}(w_{j-k})$. We can also apply this argument to $z_0 = w_{j-2k}$, $z_1 = w_{j-k}$, $z_2' = w_{j-1}$, $z_3' = w_{j+k-1}$ (since we have $z_2'/z_1 \approx \beta > \tilde{\alpha}$), thus getting $\tilde{F}^{(1)}(w_{j-1}) = \tilde{F}^{(2)}(w_{j-k})$. Combining these two equations, we get that $\tilde{F}^{(1)}(w_{j-1}) = \tilde{F}^{(1)}(w_j)$ for all $j \in \{2k, \ldots, r-k\}$. Thus,

$$\tilde{F}^{(1)}(w_{2k}) = \tilde{F}^{(1)}(w_{2k+1}) = \;\cdots\; = \tilde{F}^{(1)}(w_{r-k}) \tag{4}$$

However, assuming that $\mathsf{Hi}/\mathsf{Lo} \geq \alpha^5$, we get that $w_{r-k}/w_{2k} \geq (\mathsf{Hi}/\alpha)/(\alpha^2\mathsf{Lo}) > \tilde{\alpha}^2$, and therefore, similarly to Eq. (2), it must be that $\tilde{F}(w_k)$ and $\tilde{F}(w_{r-k})$ are disjoint, and we reach a contradiction to Eq. (4). $\square$

REMARK 5. *For the proof above, it is sufficient that the sliding-window reduction works for $\mathsf{Lo} = w_0, w_1, \ldots, w_r = \mathsf{Hi}$. Indeed, this is the way we will use Lemma 7 in the proof of Theorem 8 below.*

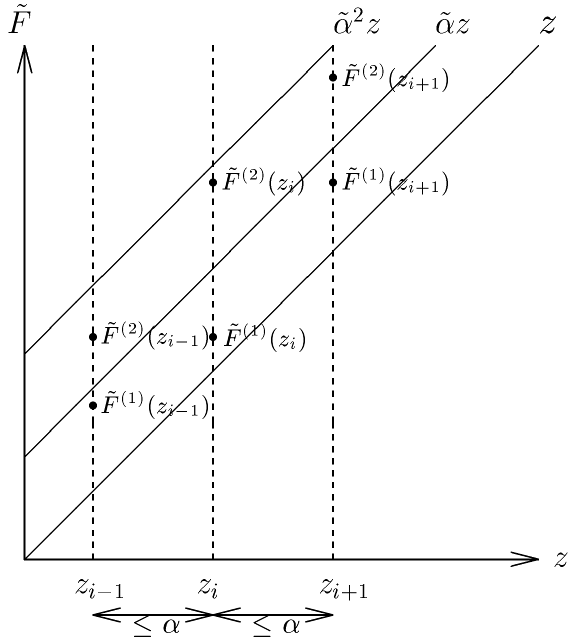

Figure 1: The structure of the values $\tilde{F}^{(j)}(z_i)$

Extensions. Lemma 7 can be extended to handle reductions that only ensure equality in one case (either $v \in L$ or $v \notin L$). This is done similarly to the way Lemma 2 is extended to handle inequality in both cases. Below we explain how to handle inequality in the case $v \in L$.

The additional property that we need is that for every $v \in L$ there is a "shift amount" $\delta_v > 0$, so that for all z and β (in the appropriate ranges), the reduction on (v, z, β) returns an instance x with $f(x) = \lceil z/(1+\delta_v) \rceil$. For technical reasons, the "shift amount" must be bounded by $1+\delta_v < \alpha/(1+\varepsilon)$ for a fixed $\varepsilon > 0$.

The only change to the proof, is that we need to show that Claim 2 remains true even for these weaker reductions. Namely, we have to show that for $z' < z'' \leq \alpha z'$, the sets $\tilde{F}_N(z'), \tilde{F}_N(z'')$ cannot be disjoint (or else we can decide the NP-hard language L). In fact, for the purpose of Lemma 7, it is sufficient to prove Claim 2 for the case where $z'' \geq \alpha z'/(1+\varepsilon)$.

So assume that we have z', z'' such that $\alpha z'/(1+\varepsilon) \leq z'' \leq \alpha z'$, and for which $\tilde{F}_N(z') \cap \tilde{F}_N(z'') = \emptyset$. We show how to modify the algorithm A from the proof of Claim 2, to decide whether $v \in L$. As before, A first computes the sets $\tilde{F}_N(z)$ for all z, and finds such values z', z''. Then, A goes over all the (polynomially many[10]) possible values of δ_v. For every possible value δ, A performs the reduction on its input v, with parameters $z = \lfloor z'(1+\delta) \rfloor$ and $\beta = z''/\lfloor z'(1+\delta) \rfloor$. (Note that $z'' \geq \lceil z' \cdot \alpha/(1+\varepsilon) \rceil > \lfloor z' \cdot (1+\delta) \rfloor$.) The reduction returns an instance x_δ of O, and A computes $F(x_\delta)$.

If $v \notin L$ then we have $f(x_\delta) = \beta z = z''$, so $F(x) \notin \tilde{F}(z')$, regardless of δ. On the other hand, if $v \in L$ then for at least one δ (namely, for $\delta = \delta_v$), we have

$$f(x) = \left\lceil \frac{\lfloor z'(1+\delta_v) \rfloor}{(1+\delta_v)} \right\rceil = z'$$

and therefore $F(x) \in \tilde{F}(z')$.

4.2.2 The minimum vertex cover problem

We exemplify the use of Lemma 7 by obtaining an inapproximability result for the minimum vertex cover problem. We note that given Håstad's $7/6-\epsilon$ (for every fixed $\epsilon > 0$) inapproximability result [12] (without privacy considerations), one could hope to get in our setting a hardness result for ratio $(7/6)^2 - \epsilon$ (for every fixed $\epsilon > 0$). Due to technical reasons, however, we only get a weaker result, proving hardness for ratio $(8/7)^2 - \epsilon$ (for every fixed $\epsilon > 0$). See details below. Recall that we defined $f_{VC}(G)$ as the minimum size of a vertex cover in G.

THEOREM 8. *For every fixed $\epsilon > 0$, the function f_{VC} has no $(8/7)^2 - \epsilon$ (polynomial time) deterministic approximation that leaks one bit of information, unless $P = NP$.*

PROOF (SKETCH). We use the extension of Lemma 7 that we described above, where there is equality only for the case $v \notin L$. Our starting point is Håstad's reduction from SAT to VC (see Theorem 8.1 in the TR version (revision 1) of [12]). On input formula ϕ, this reduction produces a graph G on $4n$ vertices, such that if $\phi \in SAT$ then the largest independent set in G is of size between $(1-\epsilon')n$ and n, and if $\phi \notin SAT$ then the largest independent set in G is of size at most $(\frac{1}{2}+\epsilon')n$. In this reduction, ϵ' is an arbitrarily small positive constant.

We first have to "fix" this reduction, to get equality for the case $\phi \notin SAT$. (This is where we lose a constant factor.) We do that by adding to the graph G an independent set of size $(\frac{1}{2}+\epsilon')n$, whose vertices are otherwise connected to all other vertices. Hence we have a graph G' on $n' = (4.5+\epsilon')n$ vertices, such that if $\phi \in SAT$ then the largest independent set in G' is of size between n and $(1-\epsilon')n$, and if $\phi \notin SAT$ then the largest independent set in G' is of size *exactly* $(\frac{1}{2}+\epsilon')n$.

Looking at the smallest vertex cover in G', we get $(3.5+\epsilon')n \leq f_{VC}(G') \leq (3.5+2\epsilon')n$ if $\phi \in SAT$, and $f_{VC}(G') = 4n$ if $\phi \notin SAT$. This gives a factor of $4/(3.5+2\epsilon') = 8/7-\epsilon''$ for an arbitrarily small fixed $\epsilon'' > 0$. We will therefore use Lemma 7 with parameter α arbitrarily close to (but smaller than) $8/7$. Note that there is a shift amount $\delta_v \geq 0$ in this reduction (since we do not have equality for $f_{VC}(G')$ in the case where $\phi \in SAT$), but we can guarantee that it is smaller than any desired constant by taking a small ϵ' (namely, sufficiently smaller than ε of Lemma 7), and hence this reduction satisfies the additional property required to handle the case where there is equality only in one case.

To get the sliding-window effect, we take $T \cdot (1+\gamma)^i$ copies of G', for $i = 0, 1, \ldots, \max$. Here, γ is a small positive constant (essentially $1+\gamma = \alpha^{1/k}$, where k is the parameter from the proof of Lemma 7). The parameter max is chosen so that $(1+\gamma)^{\max} > \alpha^5$, and T is a large enough constant so that $T(1+\gamma)^i$ is an integer for all $i \leq \max$. Note that the gap remains as before, i.e., we still have a factor of α between the case where $\phi \in SAT$ and $\phi \notin SAT$, with equality in the latter case.

[10]There are only polynomially many possible values, since we assume that $f(x)$ is an integer function, and that it is bounded between the polynomial bounds Lo and Hi.

We can "tune down" the reduction by adding a fixed graph for which f_{VC} is known: Suppose that the size of the minimum vertex cover of the graph is either at most z_1 or exactly z_2 (where $z_1 < z_2$). If we add to the graph a clique on $z_3 + 1$ vertices, then the size of the minimum vertex cover would be, respectively, either at most $z_1 + z_3$ or exactly $z_2 + z_3$. The ratio between the two cases is reduced to $(z_2 + z_3)/(z_1 + z_3) < z_2/z_1$. By choosing the value of z_3, we can reduce the ratio to all the desired values $\beta \in (1, \alpha]$. Finally, we supplement the graph with enough isolated vertices, so that we always have the same number of vertices, regardless of i and β.

It can be verified that the result is indeed a tunable sliding-window reduction (with equality in the case of $\phi \notin SAT$), with parameter $\alpha = 7/8 - \epsilon''$ and appropriate Ex, Lo and Hi. We can thus let $\tilde{\alpha}$ be a constant that is arbitrarily close to $(8/7)^2$, and the theorem follows. $\square$

5. REFERENCES

[1] T. Asano and D. P. Williamson. Improved approximation algorithms for MAX SAT. In *11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 96–105. ACM, New York, 2000.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial optimization problems and their approximability properties.* Springer Verlag, 1999. Includes a compendium of NP optimization problems, which is also available at `http://www.nada.kth.se/~viggo/wwwcompendium/`.

[3] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994.

[4] R. Bar-Yehuda, B. Chor, E. Kushilevitz, and A. Orlitsky. Privacy, additional information, and communication. *IEEE Transactions on Information Theory*, 39(6):1930–1943, 1993.

[5] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996.

[6] J. Feigenbaum, J. Fong, M. Strauss, and R. N. Wright. Secure multiparty computation of approximations. Unpublished manuscript. Presented at DIMACS Workshop on Cryptography and Intractability. March 2000.

[7] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. N. Wright. Secure multiparty computation of approximations. To appear in ICALP 2001. A longer version is available as an eprint report `http://eprint.iacr.org/2001/024`.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., 1979.

[9] O. Goldreich. Foundations of cryptography (fragments of a book). 1995. available as a monograph from the Electronic Colloquium on Computational Complexity, `http://www.eccc.uni-trier.de/eccc/`.

[10] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *27th Annual Symposium on Foundations of Computer Science*, pages 174–187. IEEE, 1986.

[11] O. Goldreich and E. Petrank. Quantifying knowledge complexity. *Computational Complexity*, 8(1):50–98, 1999. Preliminary version appeared in FOCS'91, pp. 59-68.

[12] J. Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 1–10. ACM, 1997. Also available as Report TR97-037 from ECCC, `http://www.eccc.uni-trier.de/eccc/`.

[13] D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, Boston, MA, 1997.