Gap MCSP is not (Levin) $\text{NP}$-complete in Obfustopia

Noam Mazor * Rafael Pass †

March 10, 2024

Abstract

We demonstrate that under believable cryptographic hardness assumptions, Gap versions of standard meta-complexity problems, such as the Minimum Circuit Size problem (MCSP) and the Minimum Time-Bounded Kolmogorov Complexity problem (MKTP) are not $\text{NP}$-complete w.r.t. Levin (i.e., witness-preserving many-to-one) reductions.

In more detail:

• Assuming the existence of indistinguishability obfuscation, and subexponentially-secure one-way functions, an appropriate Gap version of MCSP is not $\text{NP}$-complete under randomized Levin-reductions.

• Assuming the existence of subexponentially-secure indistinguishability obfuscation, subexponentially-secure one-way functions and injective PRGs, an appropriate Gap version of MKTP is not $\text{NP}$-complete under randomized Levin-reductions.

---

*Tel Aviv University. E-mail: noammaz@gmail.com. Research partly supported by NSF CNS-2149305 and DARPA under Agreement No. HR00110C0086.

†Tel-Aviv University and Cornell Tech. E-mail: rafaelp@tau.ac.il. Supported in part by AFOSR Award FA9550-23-1-0387, AFOSR Award FA9550-23-1-0312, and an Algorand Foundation grant. This material is based upon work supported by DARPA under Agreement No. HR00110C0086. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, DARPA, AFOSR or the Algorand Foundation.
1 Introduction

As described by Trakhtenbrot [Tra84], starting in the 1960s, there has been an on-going effort studying the computational complexity of so-called “meta-complexity” problems; notably (a) the Minimum Circuit Size problem (MCSP) [KC00; Tra84]—determining the size of the smallest Boolean circuit that computes a given function \( x \), and (b) the Time-Bounded Kolmogorov Complexity Problem (MKTP) [Kol68; Sol64; Cha69; Ko86; Har83; Sip83]—determining the the length, denoted \( K^t(x) \) of the shortest program (evaluated on some particular Universal Turing machine \( U \)) that generates a given string \( x \), within time \( t \), where \( t = poly(|x|) \) is a polynomial. In particular, a major problem since the 1960 is whether these problems, or the Gap versions of them (where the goal is to determine whether the size is above a threshold \( s_2 \) or below a threshold \( s_1 \)) are NP-complete. Indeed, as recounted by [AKRR11; Ila20; Ila23], Levin is said to have delayed the publication of his theory of NP-completeness [Lev73a] in order to show NP-completeness of MCSP.

In the following decades, there has been a lot of amazing progress—providing evidence pointing towards both a positive and a negative answer:

Towards NP-completeness: While it is still unknown whether the original problems are NP-complete, several generalizations of them have been proven to be NP-complete. Most notably, Ilango first demonstrated this for an oracle version of MCSP [Ila20]; this was subsequently extended to a multi-bit version of MCSP refering to as Multi-MCSP [ILCO20], to a conditional version of the MKTP problem, McKTP [LP22], and to other variants [Hir22a]. [HIR23] recently improved the parameters of the reduction to McKTP [LP22], assuming that witness encryption scheme exists. Additionally, Ilango [Ila23] very recently demonstrates that NP-hardness of variant of MCSP and MKTP where the programs are allowed to access a random oracle, yielding a heuristic NP-completeness Karp (i.e., many-one) reduction for these problems (if instantiating the random oracle with a concrete hash function). Finally, a recent work by Impagliazzo, Kabanets, and Volkovich [IKV23] provides various different results that can be interpreted as giving evidence that MCSP is NP-complete with respect to randomized reductions.

Towards Non NP-completeness: There is also evidence pointing towards non NP-completeness: Ko [Ko91] showed that a version of MKTP is not NP complete with respect to an oracle, and Ren and Santhanam [RS22] gave an oracle with respect to which MCSP is not NP complete. Other works prove limitations on the structure on reduction to meta-complexity problems. Murray and Williams [MW17] prove that MCSP is not NP complete under so-called local reductions. Saks and Santhanam [SS20] show that the NP-completeness of MCSP under Turing reductions with certain properties implies circuit lower bounds. For example if MCSP is complete under so-called parametric honest Turing reductions, then \( E \not\subseteq \text{SIZE}(\text{poly}) \). More recently, Saks and Santhanam [SS22] gave evidence that the running time of any randomized non-adaptive reduction from SAT to \( K^t \) approximation must grow with the time parameter \( t \). These results, however, only rule out quite limited types of reductions.

Despite this progress, the original question, however, remains wide open.
1.1 Our Results

The current paper provides strong evidence that the Gap versions of MCSP and MKTP are not NP-complete w.r.t. Levin reductions—that is witness-preserving many-to-one reductions. In particular, we demonstrate that under somewhat strong, but generally believed, cryptographic hardness assumptions, the Gap versions of MCSP is not NP-complete w.r.t. Levin reductions.

**Levin Reductions:** The three original ways [Coo71; Kar72; Lev73b] of defining NP completeness differ in how reductions from a language $L$ to a language $L'$ are defined (see e.g., [Gol08] for a discussion). Cook [Coo71] considers the most permissive notion: a Turing machine deciding $L$ having oracle access to a decider for $L'$. Karp’s notion—called a Karp reduction (or many-one reduction) is more restrictive: it requires efficiently mapping an instance $x$ into an instance $x'$ such that $x \in L$ iff $x' \in L'$. Levin’s notion, called a Levin reduction (or a witness preserving many-one reduction) is the most restrictive: it additionally requires efficiently mapping any witness $w$ for $x$ into a witness for $x'$, and furthermore any witness $w'$ for $x'$ into a witness $w$ for $x$. While Karp reductions are most commonly used, as far as we are aware, most natural NP-completeness reduction are actually of the Levin type as well. Furthermore, for constructive applications of NP-completeness, NP-hardness demonstrated using a Levin reduction is typically what is needed: In particular, for cryptographic application to interactive proofs (e.g., demonstrating that every language in NP has a zero-knowledge proof of knowledge [FFS87], or that every languages in NP has a succinct argument [BG09], the notion of a Levin reduction is crucial (see e.g., [BG09] that in particular notes that even the most sophisticated NP completeness reductions, as those provided by the PCP theorem [ALMSS98; AS98], are Levin reductions). Our focus here are on such Levin reductions; in particular, we will present the (conditional) impossibility of Levin reductions for demonstrating NP-completeness; in fact, our impossibility will apply not only to deterministic but also randomized Levin reductions (where the reduction is allowed to fail with some small constant probability).

We mention that e.g., the NP-completeness results of [Ila23] and [LP22] rely on the NP-completeness of approximation for the Set-Cover problem [DGKR03; Tre01]. In both works, the reductions from Set-Cover to the GapMCSP and GapMKP (or the conditional version in the case of [LP22]) are (randomized) Levin reductions (see Appendix A for a discussion of the result of [Ila23]). The Set-cover NP completeness itself relies on a long sequence of the reductions that we have not been able to verify whether they are all Levin (although, as mentioned above, the main technical core, the PCP theorem, is).

**Our Cryptographic Hardness Assumptions: Indistinguishability Obfuscation:** We will rely on the existence of indistinguishability obfuscation (iO) for circuits [Bar+01]. Roughly speaking, an indistinguishability obfuscator is an efficient algorithms $iO$ that given circuit $C$ outputs an “obfuscated” version of $C$ having the property that obfuscations of any two functionally equivalent circuits are indistinguishable. Following the ground-breaking work of [Gar+16], several heuristic candidates were proposed, as well as provably secure constructions based on various assumptions [PST14; GLSW15; Lin16; WW21; LT17; LV16; Lin17; AJS18; JLMS19; JAMS19; AJLMS19; GJLS21; APM20; Agr19]. Most notable, the recent breakthrough result presents a construction based on several well-founded (and generally believed) hardness assumption [JLS21]. (Constructions based on less standard, but seemingly quantum-safe, “circular-security” assumptions also appear in [BDGM23; GP21; BDGM20]).
For our main results on MCSP, we will simply rely on indistinguishability obfuscation and subexponentially-secure one-way function. For our results on MKTP, we will rely on $iO$ with subexponential security as well as other standard cryptographic assumptions such as injective pseudorandom generators (PRGs), that e.g., are implied by the existence of one-way permutations.

Main Theorem  We present the following main result:

- Assuming the existence of indistinguishability obfuscation and subexponentially-secure one-way function, an appropriate Gap version of MCSP is not $\text{NP}$-complete under randomized Levin-reductions.

- Assuming the existence of subexponentially-secure indistinguishability obfuscation, subexponentially-secure one-way function and injective PRGs, an appropriate Gap version of MKTP is not $\text{NP}$-complete under randomized Levin-reductions.

In more detail, let $\text{GapMCSP}[s_0, s_1]$ be the promise problem in which given a truth table $x$ we need to distinguish between the following two cases:

- **Yes instances**: There exists a circuit $C$ of size at most $s_0(|x|)$ that computes $x$.

- **No Instances**: There is no circuit of size $s_1(|x|)$ that computes $x$.

Our first theorem states that when the gap between $s_0$ and $s_1$ is large enough, and under cryptographic assumptions, $\text{GapMCSP}[s_0, s_1]$ is not $\text{NP}$-complete with respect to Levin reductions.

**Theorem 1.1.** Assume that iO and subexponentially-secure one-way function exist. Then there exists a polynomial $p$, such that for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n))$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not $\text{NP}$ complete with respect to Levin reductions.

We remark that if all of the assumed cryptographic primitives are secure against sub-exponential adversaries (in contrast to just polynomial adversaries), then our results rule out also randomized Levin reductions that run in sub exponential time.

Additionally, the assumption of subexponentially-secure one-way function in Theorem 1.1 is only to handle so-called non honest reductions: A Karp reduction $f$ is to be honest if for every $x \in \{0,1\}^*$, $|f(x)| \geq |x|^\delta$ for some constant $\delta > 0$ (i.e., the mapping from statements $x$ to $x'$ is polynomially preserving).

To exclude only honest reductions, it is enough to assume one-way function with polynomial security. Such one-way functions are known to exist assuming iO and the minimal assumption that $\textbf{P} \notin \text{ioBPP}$ [Kom+14]. We get the following theorem.

**Theorem 1.2.** Assume that iO exists, and that $\textbf{P} \notin \text{ioBPP}$. Then there exists a polynomial $p$, such that for every $\epsilon > 0$, for any pair of efficiently computable functions $s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n))$ and $s_0(n) > n^\epsilon$, it holds that $\text{GapMCSP}[s_0(n), s_1(n)]$ is not $\text{NP}$ complete with respect to honest Levin reductions.

Our second result is a similar result for the $\text{Gap}_{\text{p}}\text{MK}^tP$ problem. Recall that $K^t(x)$ is the minimal length of a program that outputs $x$ within $t(|x|)$ steps. For polynomials $t$ and $p$, let $\text{Gap}_{\text{p}}\text{MK}^tP[s_0, s_1]$ be the promise problem in which given a string $x$ we need to distinguish between the following two cases:
• Yes instances: $K^t(x) \leq s_0(|x|)$

• No Instances: $K^{p(t)}(x) > s_1(|x|)$.

We prove the following theorem.

**Theorem 1.3.** Assume that subexponentially-secure iO, subexponentially-secure one-way functions and injective PRG exist. Then there exist a polynomial $q$ such that for any $t \in \text{poly}$ and any efficiently computable functions $s_0, s_1 : \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > q(\log t(n), s_0(n))$, and for every large enough polynomial $p$, it holds that $\text{Gap}_{p}\text{MK}^1\text{P}[s_0, s_1]$ is not NP complete with respect to Levin reductions.

**Achieving a smaller gap under stronger assumptions** As discussed above, several generalization of the GapMCSP and Gap$_p$MKTP problem have been proven NP complete. The work of [Ila23] showed that the same problems we consider here are NP relative to a random oracle. There, the gap between the Yes and No instances is a multiplicative $(1+\epsilon)$ gap, for a small constant $\epsilon > 0$ while in the theorems above we need the gap to be larger. Similarly, [LP22] showed that deciding a conditional version of MKTP is NP-hard, and their result can be generalized to a gap problem with a larger constant multiplicative factor. Hirahara [Hir22b] used a reduction from the Minimum Monotone Satisfying Assignment problem to McKTP, resulting with a NP-hardness of the GapMcKTP with larger multiplicative gap, but still sub polynomial in the input length ($n^{o(1)}$).

The polynomial $p$ in Theorems 1.1 and 1.2 is the overhead of the iO algorithm we use. By assuming a stronger assumption—that iO with a small overhead exists—we can improve the gap. For example, we say that iO has additive overhead if on input $C$ and security parameter $\lambda$, the size of the obfuscated circuit is $|C| + \text{poly}(\lambda)$. If we assume iO with additive overhead, we would get hardness of GapMCSP also for the additive gap case. Unfortunately, such iO constructions are currently not know (but as far as we know, there are also no results indicating that this should be impossible). However, if we consider slightly stronger assumptions, we can get iO with a factor $2 + \epsilon$ overhead (for any constant $\epsilon > 0$) [AJS17], yielding the following theorems:

**Theorem 1.4.** Assume subexponential-secure iO, and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every constant $\epsilon > 0$, and for any efficiently computable function $s_0$, it holds that $\text{GapMCSP}[s_0(n), (2 + \epsilon)s_0(n)]$ is not NP complete with respect to Levin reductions.

**Theorem 1.5.** Assume subexponential-secure iO, and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every very constant $\epsilon > 0$, for every large enough polynomial $p$, and for every efficiently computable function $s_0$ it holds that $\text{Gap}_{p}\text{MK}^1\text{P}[s_0, (2+ \epsilon)s_0(n)]$ is not NP complete with respect to Levin reductions.

**Proof Overview** In this proof outline, we will for simplicity focus on ruling out deterministic Levin reductions for the GapMCSP problem. Additionally, on top of the existence of iO, we will here assume the existence of collision-resistant hash function; that is the existence of a family of compressing functions such that for a randomly sampled $h$, it is infeasibly to find two inputs $x_1, x_2$ that “collide” (i.e., $h(x_1) = h(x_2)$) although such collision exists. (In our actual proof, we instead rely on the weaker primitive of a target collision-resistant hash function (TCR; also known as, universal one-way hash function [NY89]) which can be constructed from one-way functions.
Given a witness $w_1$ such that $h(w_1) = x$, we use the Levin reduction to get MCSP witness. Then we use the iO to get a new MCSP witness, and use the Levin reduction again to get back $\tilde{w}_1$ such that $h(\tilde{w}_1) = x$.

Figure 1: The proof overview. Given a witness $w_1$ such that $h(w_1) = x$, we use the Levin reduction to get MCSP witness. Then we use the iO to get a new MCSP witness, and use the Levin reduction again to get back $\tilde{w}_1$ such that $h(\tilde{w}_1) = x$.

[Rom90]. Finally, let us start by assuming that the reduction is "honest" (i.e., mapping statements $x$ to statements $x'$ of polynomially-related length.

The key idea will be to use the Levin reduction and the iO in order to find a collision for $h$. Roughly speaking, we start by sampling some $w_1$ and compute $x = h(w_1)$; we think of $x$ as a statement for the language of images of $h$, and of $w_1$ as the witness for $x$. We next use the Levin reduction to get an MCSP statement $x'$ and its corresponding witness $w'_1$. Note that the witness $w'_1$ is a circuit computing $x'$. We then obfuscate $w'_1$ using the iO to get a new witness $\tilde{w}'_1$ for $x'$. Using the Levin reduction, we can finally turn $\tilde{w}'_1$ into a (hopefully new) witness $\tilde{w}_1$ for $x$. Indeed, the key point is that if we had started with a different preimage $w_2 \neq w_1$ for $x = h(w_1)$ and done the same process, then $w'_2$ would become a functionally equivalent circuit to $w'_1$ and thus by the security of the iO, the distributions of $\tilde{w}'_2$ and $\tilde{w}'_1$ are computationally indistinguishable, so we conclude that $\tilde{w}_2$ and $\tilde{w}_1$ also are. In particular, it follows that $\tilde{w} \neq w$ with probability at least $1/2$, and we have thus found a collision.

Note that we here rely on the NP-completeness of the Gap version of the MCSP problem since when applying the iO we get a new witness for $x'$ but this witness (i.e., the circuit) is bigger than the original one. In particular, the overhead of the iO translates into the gap of the problem—for instance, if the overhead of the iO is only linear, we can handle a linear gap, and if it has polynomial overhead then we can only rule out reductions for the polynomial gap version of the problem.

Dealing with Non-honest Reductions If the reduction is not honest, the statement $x'$ could be a lot shorter than $x$; the problem then becomes if we run the iO on a security parameter that is polynomially related to $|x'|$ (which we require to ensure that we stay within the promise), we may no longer have security with respect to an attacker who runs in time polynomial in $|x| = n$ (which is required to ensure that we find a collision). However, if we start of with a collision-resistant hash function with sub-exponential security (i.e., $2^{n^\epsilon}$ security), we can resolve this problem using a case-analysis. If $|x'| \leq n^\epsilon$, then we simply find a new witness $\tilde{w}'$ using brute-force search, and otherwise use the iO. This ensures that we only run the iO in case the reduction behaves "honestly"; on the other hand, when the reduction chooses a short $x'$, we still contradict the subexponential security of the collision-resistant hash function.

Extensions for GapMKP. We next generalize the above proof for the GapMKP problem. To be able to do so, we need a way to move from one GapMKP witness to another, when a GapMKP witness is a $t$-time TM $P$ of size at most $s_0(|x|)$ that outputs $x$. A naive approach is to first convert the TM $P$ into a circuit, then apply the iO for circuits, and lastly, convert the
circuit back to a TM. The problem in this approach is that since the program \( P \) outputs \( x \), the time bound \( t \) must be at least \( |x| \). This means that the circuit we construct from \( P \) will have trivial size, and we will not be able to get back a non-trivial program that outputs \( x \).

Luckily, we can use \( iO \) for TMs directly on \( P \), or even it suffices to rely on a weaker primitive of a randomized encoding. Randomized encoding for TMs are known to exist assuming subexponential-secure \( iO \) for circuits and injective PRGs [KLW15; LPST15].

**Discussion.** The results presented yield give a strong evidence that the GapMCSP and Gap\(_p\)MK\(_t\)P are not \( \text{NP} \)-complete w.r.t. Levin reductions, at least when the gap is at least a factor 2. Furthermore, although there are no known constructions of \( iO \) with only additive overhead based on well-founded hardness assumptions, one can come up with candidate constructions with only linear additive overhead and heuristically assume that they satisfy the notion of indistinguishability obfuscation.\(^1\) Under these more heuristic assumptions (which in our eyes seem reasonable), our results thus give evidence that these problems are not \( \text{NP} \)-complete w.r.t. Levin reductions even when the gap is a small additive term. These results thus provide (in our eyes) convincing cryptographic evidence that the original task set out by Levin is impossible (since he indeed defined \( \text{NP} \)-completeness through the notion of what today is referred to as a Levin reduction.)

Of course, it could still be that a weaker notion of a reduction (e.g., a Karp) reduction can be used to prove \( \text{NP} \)-completeness of these problems. In particular, consider the results of [Ila23], which shows \( \text{NP} \)-completeness of GapMCSP in the random oracle model. While, as discussed, his reduction from (approximate) Set-Cover to GapMCSP is a Levin reductions (see Appendix A), the witness preserving part of the reduction relies on the random oracle—in particular, the witness reconstruction step relies on observing the queries to the random oracle performed by the circuit \( \tilde{w}' \) (i.e., the witness for the transformed statement \( x' \)). If instantiating the random oracle with a concrete hashfunction \( h \), it is no longer clear how to perform this task—in particular if the circuit has be obfuscated so that it (intuitively) becomes hard to find the code of \( h \) in the description of the circuit. As such, when instantiating the random oracle with a hashfunction, the reduction most likely is no longer a Levin reduction, but conceivably it could still be a Karp reduction.

## 2 Preliminaries

### 2.1 Notations

All logarithms are taken in base 2. We use calligraphic letters to denote sets and distributions, uppercase for random variables, and lowercase for values and functions. Given a set \( S \subseteq \{0, 1\}^* \), we let \( \bar{S} = \{0, 1\}^* \setminus S \). Let \( \text{poly} \) stand for the set of all polynomials. Let \( \text{ppt} \) stand for probabilistic \( \text{poly-time} \), and \( \text{n.u.-poly-time} \) stand for non-uniform \( \text{poly-time} \). An \( \text{n.u.-poly-time} \) algorithm \( A \) is equipped with a (fixed) poly-size advice string set \( \{z_n\}_{n \in \mathbb{N}} \). Let \( \text{neg} \) stand for a negligible function. For a SAT formula \( \phi \) over \( n \) variables and an assignment \( v \in \{0, 1\}^n \), we use \( \phi[v] \in \{0, 1\} \) to denote the truth value of the evaluation of \( \phi \) on \( v \).

\(^1\)In particular, take the constructions from e.g., [BCP14; AJS17] and instead of encrypting the program twice under an FHE with additive linear overhead, simply encrypt the program once. While the two encryptions are needed for the security proof, the construction without the two encryptions seems heuristically secure.
2.2 Distributions and Random Variables

When unambiguous, we will naturally view a random variable as its marginal distribution. The support of a finite distribution \( \mathcal{P} \) is defined by \( \text{Supp}(\mathcal{P}) := \{ x : \Pr_{\mathcal{P}}[x] > 0 \} \). For a (discrete) distribution \( \mathcal{P} \), let \( x \leftarrow \mathcal{P} \) denote that \( x \) was sampled according to \( \mathcal{P} \). Similarly, for a set \( S \), let \( x \leftarrow S \) denote that \( x \) is drawn uniformly from \( S \).

2.3 Kolmogorov Complexity

Roughly speaking, the \( t \)-time-bounded Kolmogorov complexity, \( K^t(x) \), of a string \( x \in \{0,1\}^* \) is the length of the shortest program \( \Pi = (M,y) \) such that, when simulated by an universal Turing machine, \( \Pi \) outputs \( x \) in \( t(|x|) \) steps. Here, a program \( \Pi \) is simply a pair of a Turing Machine \( M \) and an input \( y \), where the output of \( \Pi \) is defined as the output of \( M(y) \). When there is no running time bound (i.e., the program can run in an arbitrary number of steps), we obtain the notion of Kolmogorov complexity.

In the following, let \( U(\Pi, 1^t) \) denote the output of \( \Pi \) when emulated on \( U \) for \( t \) steps. We now define the notion of Kolmogorov complexity with respect to the universal TM \( U \).

**Definition 2.1.** Let \( t \in \mathbb{N} \) be a number. For all \( x \in \{0,1\}^* \), define

\[
K_U^t(x) = \min_{\Pi \in \{0,1\}^*} \{|\Pi| : U(\Pi, 1^t) = x\}
\]

where \( |\Pi| \) is referred to as the description length of \( \Pi \).

It is well known that for every \( x \), \( K^t(x) \leq |x| + c \), for some constant \( c \) depending only on the choice of the universal TM \( U \).

**Fact 2.2.** For every universal TM \( U \), there exists a constant \( c \) such that for every \( x \in \{0,1\}^* \), and for every \( t \) such that \( t(n) > 0 \), \( K_U^t(x) \leq |x| + c \).

2.4 Levin Reductions

For a relation \( \mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^* \), let \( \mathcal{L}(\mathcal{R}) = \{ x \in \{0,1\}^* : \exists w \in \{0,1\}^* \text{ s.t. } (x,w) \in \mathcal{R} \} \). We say that a relation \( \mathcal{R} \) is the witness relation of a language \( \mathcal{L} \subseteq \{0,1\}^* \) if \( \mathcal{L}(\mathcal{R}) = \mathcal{L} \).

**Definition 2.3** (Levin reduction). Let \( \mathcal{R}_1 \) and \( \mathcal{R}_2 \) be relations. A triplet of efficiently computable functions \((f,g,h)\) is a Levin reduction from \( \mathcal{R}_1 \) to \( \mathcal{R}_2 \) if

\[
\begin{align*}
\bullet & \text{ For every } (x,w) \in \mathcal{R}_1, (f(x),g(x,w)) \in \mathcal{R}_2. \\
\bullet & \text{ If } (f(x),w) \in \mathcal{R}_2 \text{ then } (x,h(x,w)) \in \mathcal{R}_1.
\end{align*}
\]

**Remark 2.4.** Notice that if \((f,g,h)\) a Levin reduction from \( \mathcal{R}_1 \) to \( \mathcal{R}_2 \), then \( f \) is a Karp reduction from \( \mathcal{L}(\mathcal{R}_1) \) to \( \mathcal{L}(\mathcal{R}_2) \). Indeed, the first item above implies that if \( x \in \mathcal{L}(\mathcal{R}_1) \) then \( f(x) \in \mathcal{L}(\mathcal{R}_2) \), and the second item implies the other direction.

A Levin reduction \((f,g,h)\) is honest if there exists a constant \( \delta > 0 \) such that for every large enough \( n \in \mathbb{N} \) and every \( x \in \{0,1\}^n \), \( f(x) \geq n^\delta \).

When for two languages \( \mathcal{L}_1 \) and \( \mathcal{L}_2 \) we fix canonical relations \( \mathcal{R}_1 \) and \( \mathcal{R}_2 \), we say that there is a Levin reduction from \( \mathcal{L}_1 \) to \( \mathcal{L}_2 \) if there is a Levin reduction from \( \mathcal{R}_1 \) to \( \mathcal{R}_2 \). We say that \( \mathcal{L} \in \text{NP} \).
is \( \mathsf{NP} \) complete under Levin reductions if there exists a Levin reduction from SAT to \( \mathcal{L} \), where the canonical relation for SAT is

\[
\mathcal{R}_{\text{SAT}} = \{(\phi, v) : \phi \text{ is a SAT formula and } \phi[v] = 1\}.
\]

We also define Levin reductions for promise problems. In the following, we consider promise problem \( (\mathcal{Y}, \mathcal{N}) \) that associated with two relations \( (\mathcal{R}_\mathcal{Y}, \mathcal{R}_\mathcal{N}) \) such that \( \mathcal{R}_\mathcal{Y} \subseteq \mathcal{R}_\mathcal{N} \), where \( \mathcal{R}_\mathcal{Y} \) is the witness relation for \( \mathcal{Y} \), and \( \mathcal{R}_\mathcal{N} \) is the witness relation for \( \mathcal{N} \). That is, \( (\mathcal{Y}, \mathcal{N}) = (\mathcal{L}(\mathcal{R}_\mathcal{Y}), \mathcal{L}(\mathcal{R}_\mathcal{N})) \).

**Definition 2.5** (Levin reduction, promise problems). Let \( (\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\mathcal{N}}^1) \) and \( (\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\mathcal{N}}^2) \) be pairs of relations such that \( \mathcal{R}_{\mathcal{Y}}^1 \subseteq \mathcal{R}_{\mathcal{N}}^1 \) and \( \mathcal{R}_{\mathcal{Y}}^2 \subseteq \mathcal{R}_{\mathcal{N}}^2 \). A triplet of efficiently computable functions \( (f, g, h) \) is a Levin reduction from \( (\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\mathcal{N}}^1) \) to \( (\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\mathcal{N}}^2) \) if

- For every \( (x, w) \in \mathcal{R}_{\mathcal{Y}}^1 \), \( (f(x), g(x, w)) \in \mathcal{R}_{\mathcal{N}}^2 \).
- If \( (f(x), w) \in \mathcal{R}_{\mathcal{N}}^2 \) then \( (x, h(x, w)) \in \mathcal{R}_{\mathcal{N}}^1 \).

Note that we can define reductions from language to promise problem by taking \( \mathcal{R}_\mathcal{Y} = \mathcal{R}_\mathcal{N} \). Lastly, our results hold even when the reductions allowed to be randomized. In this case, \( f(x; r) \) can be randomized function (that uses randomness \( r \)), and both \( g, h \) get access to \( r \) (and possibly use more randomness). We then only require that the above requirements hold with high probability over \( r \).

**Definition 2.6** (Randomized Levin reduction, promise problems). Let \( (\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\mathcal{N}}^1) \) and \( (\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\mathcal{N}}^2) \) be pairs of relations such that \( \mathcal{R}_{\mathcal{Y}}^1 \subseteq \mathcal{R}_{\mathcal{N}}^1 \) and \( \mathcal{R}_{\mathcal{Y}}^2 \subseteq \mathcal{R}_{\mathcal{N}}^2 \). A triplet of efficiently computable functions \( (f, g, h) \) is a randomized Levin reduction with \( \epsilon \)-error from \( (\mathcal{R}_{\mathcal{Y}}^1, \mathcal{R}_{\mathcal{N}}^1) \) to \( (\mathcal{R}_{\mathcal{Y}}^2, \mathcal{R}_{\mathcal{N}}^2) \) if

- For every \( x \in \mathcal{L}(\mathcal{R}_{\mathcal{Y}}^1) \), with probability at least \( 1 - \epsilon \) over the choice of \( r_1 \) the following holds:
  1. \( (f(x; r_1), g(x, w; r_1)) \in \mathcal{R}_{\mathcal{N}}^2 \), and,
  2. for every \( w' \) such that \( (f(x; r_1), w') \in \mathcal{R}_{\mathcal{N}}^2 \) it holds that

\[
\Pr_{r_2 \leftarrow \{0, 1\}^*}[\{x, h(x, w'; r_1, r_2)\} \in \mathcal{R}_{\mathcal{N}}^1] \geq 1 - \epsilon.
\]

- For every \( x \notin \mathcal{L}(\mathcal{R}_{\mathcal{Y}}^1) \) it holds that \( \Pr_{r_1 \leftarrow \{0, 1\}^*}[f(x; r_1) \in \mathcal{L}(\mathcal{R}_{\mathcal{N}}^2)] \leq \epsilon.\)

### 2.5 Cryptographic Primitives

In this part we define the cryptographic tools we will use. We start with the definition of one-way function.

**Definition 2.7** (One-way function). A polynomial-time computable function \( f : \{0, 1\}^* \rightarrow \{0, 1\}^* \) is called a **one-way function** if for every PPT algorithm \( A \), there is a negligible function \( \mu : \mathbb{N} \rightarrow [0, 1] \) such that for every \( n \in \mathbb{N} \)

\[
\Pr_{x \leftarrow \{0, 1\}^n}[A(f(x)) \in f^{-1}(f(x))] \leq \mu(n).
\]

A one-way function is **subexponentially-secure** if there exists a constant \( \delta > 0 \) such that for every \( 2^{n^\delta} \) time algorithm \( A \), and for every large enough \( n \in \mathbb{N} \)

\[
\Pr_{x \leftarrow \{0, 1\}^n}[A(f(x)) \in f^{-1}(f(x))] \leq 2^{-n^\delta}.
\]
Next, we define iO.

**Definition 2.8** (Indistinguishability obfuscation). An efficiently randomized algorithm iO is an indistinguishability obfuscator if for every \( \lambda, n \in \mathbb{N} \) and any circuit \( C : \{0,1\}^n \rightarrow \{0,1\} \),

\[
\Pr_{\hat{C} \leftarrow \text{iO}(1^\lambda, C), x \leftarrow \{0,1\}^n} \left[ C(x) = \hat{C}(x) \right] = 1,
\]

and for every \( s \in \text{poly} \) and every \( n.u.-\text{poly-time algorithm} \) \( A \), there exists a negligible function \( \mu \), such that for every \( \lambda \in \mathbb{N} \) and every two circuit \( C, C' : \{0,1\}^n \rightarrow \{0,1\} \) with \( |C| = |C'| \leq s(\lambda) \) and \( n \leq \lambda \),

\[
\left| \Pr[A(1^\lambda, iO(1^\lambda, C)) = 1] - \Pr[A(1^\lambda, iO(1^\lambda, C')) = 1] \right| \leq \mu(\lambda).
\]

We say that iO has overhead \( p \) if for every \( C \) and \( \lambda \), \( |iO(1^\lambda, C)| \leq p(|C|, \lambda) \) with probability 1.

Next we define Target collision resistant hash functions, also known as universal one-way hash functions.

**Definition 2.9** (Target collision resistant hash). An efficiently computable function \( T : \{0,1\}^n \rightarrow \{0,1\}^{n-s(n)} \) is a Target collision resistant hash function (TCR) if \( s(n) \geq 1 \) and for every \( \text{PPT algorithm} \) \( A \),

\[
\Pr_{x \leftarrow \{0,1\}^n} \left[ x' \leftarrow A(x); T(x) = T(x') \text{ and } x \neq x' \right] = \text{neg}(n).
\]

We say that a TCR is secure against subexponential adversaries if there exists a constant \( \delta > 0 \) such that for every \( 2^n^\delta \text{ time algorithm} \) \( A \),

\[
\Pr_{x \leftarrow \{0,1\}^n} \left[ x' \leftarrow A(x); T(x) = T(x') \text{ and } x \neq x' \right] = \text{neg}(n).
\]

Rompel [Rom90] showed that TCR can be constructed from one-way functions.

**Theorem 2.10** ([Rom90]). Assume that one-way functions exist. Then TCR \( T : \{0,1\}^n \rightarrow \{0,1\}^{n-s(n)} \) with \( s(n) \in \omega(\log n) \) exists.

Since the proof above is black-box, the same holds for subexponential adversaries.

**Theorem 2.11**. Assume that subexponentially-secure one-way functions exist. Then there exists a TCR \( T : \{0,1\}^n \rightarrow \{0,1\}^{n-s(n)} \) secure against subexponential adversaries, with \( s(n) \in \omega(\log n) \).

We will also use the following theorem, by Komargodski et al. [Kom+14].

**Theorem 2.12** ([Kom+14]). Assume that iO exists and \( \text{P} \not\subseteq \text{ioBPP} \). Then one-way functions exist.

Lastly, we will also use the fact that a TCR is a one-way function.

**Claim 2.13.** Let \( T : \{0,1\}^n \rightarrow \{0,1\}^{n-s(n)} \) be a TCR with \( s(n) \in \omega(\log n) \). Then \( T \) is a one-way function. That is, for every \( \text{PPT algorithm} \) \( A \),

\[
\Pr_{x \leftarrow \{0,1\}^n} \left[ A(f(x)) \in T^{-1}(T(x)) \right] = \text{neg}(n).
\]

Moreover, if secure against subexponential adversaries, the above holds for any algorithm \( A \) with running time at most \( 2^n^\delta \), for some constant \( \delta \).
We sketch the proof here.

Proof. Assume that algorithm $A$ can invert $T$ with non-negligible probability. We claim that $A$ can be used to find a collision with non-negligible probability. Indeed, let $X \leftarrow \{0,1\}^n$ be a uniformly distributed random variable. Let $A'$ be the algorithm that given random input $X$, execute $A(T(X))$ and outputs its output.

Given that $A(T(X))$ found a pre-image $x'$ of $T(X)$, we get that the input of $A'$, $X$, uniformly distributed over the set $T^{-1}(T(x'))$. Since the size of $T^{-1}(T(x'))$ is large (the probability that $|T^{-1}(T(x'))| \leq k$ is at most $k \cdot 2^{-s(n)}$), with high probability it holds that $x \neq X$, and thus $A'$ found a collision.

3 GapMCSP is not NP-complete under Levin Reductions

In this section we prove our main result for GapMCSP. We first define GapMCSP[$s_0, s_1$]. In the following, a circuit $C$ computes a string $x$ if the truth table of $C$ is $x$.

Definition 3.1. For two functions $s_0, s_1 : \mathbb{N} \rightarrow \mathbb{N}$, let GapMCSP[$s_0, s_1$] denote the following promise problem.

- $\mathcal{Y} = \{x \in \{0,1\}^n : \text{There exists a circuit } C \text{ of size at most } s_0(n) \text{ that computes } x\}$
- $\mathcal{N} = \{x \in \{0,1\}^n : \text{There is no circuit of size } s_1(n) \text{ that computes } x\}$

We define the relations $\mathcal{R}_Y$ and $\mathcal{R}_N$ for GapMCSP[$s_0, s_1$] in the natural way:
\[
\mathcal{R}_Y = \{(x,C) : C \text{ is a circuit of size at most } s_0(n) \text{ that computes } x\},
\]
and,
\[
\mathcal{R}_N = \{(x,C) : C \text{ is a circuit of size at most } s_1(n) \text{ that computes } x\}.
\]

We start with the following theorem for deterministic reductions. In Section 3.2 we prove a similar theorem for randomized Levin reductions.

Theorem 3.2. Let $p : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be a function. Assume that there exist $iO$ with overhead $p$, and subexponentially-secure one-way function. Then for any constant $\alpha > 0$ and for any pair of efficiently computable functions $s_0, s_1 : \mathbb{N} \rightarrow \mathbb{N}$ for which $s_1(n) > p(s_0(n), (s_0(n))^\alpha)$, it holds that GapMCSP[$s_0(n), s_1(n)$] is not NP complete with respect to Levin reductions.

Since $iO$ is an efficient algorithm, the overhead of any $iO$ is polynomial. Combining this observation with Theorem 3.2 yields Theorem 1.1. [AJS17] construct $iO$ with multiplicative overhead.

Theorem 3.3 ([AJS17]). Assuming subexponentially-secure $iO$ and subexponentially secure re-randomizable encryption schemes, there exists an $iO$ with overhead $p(|C|, \lambda) = 2|C| + \text{poly}(\lambda)$.

Re-randomizable encryption schemes can be constructed from subexponential LWE and DDH. By taking $\alpha$ to be small enough, we get the following corollary.

Corollary 3.4. Assume subexponential-secure $iO$, and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every constant $\alpha > 0$, and for any efficiently computable function $s_0$, it holds that GapMCSP[$s_0(n), (2 + \alpha)s_0(n)$] is not NP complete with respect to Levin reductions.
3.1 Proving Theorem 3.2

To prove Theorem 3.2, let $iO$ be an indistinguishability obfuscator, and let $p \in \text{poly}$ be the overhead of $iO$. Let $T: \{0,1\}^n \rightarrow \{0,1\}^{n-o(\log n)}$ be a TCR with security against subexponential algorithms.

Consider the following distribution ensemble $D = \{D_n\}_{n \in \mathbb{N}}$ over SAT formulas and assignments $(\phi, v)$. For every $n \in \mathbb{N}$, to sample from $D_n$: sample a random $x \in \{0,1\}^n$. Let $\phi_{T(x)}$ be a formula such that $\phi_{T(x)}[x] = 1$ if and only if $T(x') = T(x)$. Output $(\phi_{T(x)}, x)$. We remark that $\phi_{T(x)}$ only depends on the value of $T(x)$ and not on $x$ itself.

We start with the following claim.

**Claim 3.5.** The following hold for every $n \in \mathbb{N}$:

- $\Pr_{(\phi,v) \leftarrow D_n}[\phi[v] = 1] = 1$
- $\Pr_{(\phi,v) \leftarrow D_n}[\exists v' \text{ s.t. } v \neq v' \text{ and } \phi[v'] = 1] = 1 - \text{neg}(n)$, and,
- for every PPT algorithm $A$

\[
\Pr_{(\phi,v) \leftarrow D_n}[A(\phi, v) = v'; v \neq v' \text{ and } \phi[v'] = 1] = \text{neg}(n).
\]

**Proof.** The first and last items follow directly from the definition of the distribution $D$ and the definition of TCR. The second item holds since $T$ is shrinking.

We also prove the following claim, that states that for any reduction $f$ from SAT to GapMCSP, the output of $f$ on inputs samples from $D_n$ must length polynomial in $n$. Here we need the subexponential security of $T$.

**Claim 3.6.** Let $(f,g,h)$ be a Levin reduction from SAT to GapMCSP[$s_0, s_1$]. Then there exists a constant $\delta > 0$ such that

\[
\Pr_{(\phi,v) \leftarrow D_n}[s_0(|f(\phi)|) \geq n^{\delta}] \geq 1 - \text{neg}(n).
\]

**Remark 3.7.** Claim 3.6 is the only place in which we use the subexponential security assumption. We need it to make sure that (with high probability over $D$) $|s_0(f(\phi))|$ is not too small. While we can require that $s_0(n) \geq n^\epsilon$ for some $\epsilon > 0$, the reduction $f$ itself can return short outputs.

When the reduction $f$ is honest (that is, $|f(x)| \geq |x|^{\alpha}$ for all inputs $x$ and for some $\alpha > 0$), we can replace the assumption on exponentially-secure one-way function with the above requirement that $s_0(n) \geq n^\epsilon$, and minimal assumption that $P \not\subseteq \text{ioBPP}$. The latter assumption is known to imply (together with $iO$) one-way function (see Theorem 2.12). Using the same proof as follows we get Theorem 1.2.

**Proof.** Assume toward a contradiction that this is not the case for all constant $\delta > 0$. We will show how to invert $T$. That is, we will show an algorithm $A$ that runs in time $2^{n^{c-\delta}}$ for some constant $c$ such that

\[
\Pr_{x \leftarrow \{0,1\}^n}[A(T(x)) \in T^{-1}(T(x))] \geq \Pr_{(\phi,v) \leftarrow D_n}[s_0(|f(\phi)|) < n^{\delta}].
\]

The claim will then follow by Claim 2.13, as by assumption $\Pr_{(\phi,v) \leftarrow D_n}[s_0(|f(\phi)|) < n^{\delta}]$ is noticeable for all choices of $\delta > 0$ (and for infinitely many $n$’s).
Let $A$ be the algorithm that given $y = T(x)$, construct the formula $\phi_y$, and computes $f(\phi_y)$. Then, if $s_0(\|f(\phi_T(x))\|) < n^{\Delta}$, $A$ uses brute force to find a circuit $C$ of size at most $s_0(\|f(\phi_y)\|)$ that computes $f(\phi_y)$. Lastly, $A$ outputs $h(\phi_y, C)$.

It is not hard to see that $A$ runs in time $2^{\text{poly}(n^{\Delta})}$. By the definition of Levin reductions, when $s_0(\|f(\phi_T(x))\|) < n^{\Delta}$, $A$ always outputs $x'$ such that $T(x') = T(x)$. Lastly, observe that the distribution of $\phi_y$ for $y = T(x)$ when $x \leftarrow \{0,1\}^n$, is exactly the distribution of $\phi$ when $(\phi, v) \leftarrow D_n$.

The next lemma shows it is possible to use iO to find collisions in the TCR.

**Lemma 3.8.** Let iO be indistinguishability obfuscator with overhead $p$, and let $s_0$ and $s_1$ as in Theorem 3.2. Assume that there exists a Levin reduction from SAT to GapMCSP[$s_0, s_1$]. Then there exists an efficient algorithm $A$ such that for every large enough $n \in \mathbb{N}$

$$\Pr_{(\phi, v) \leftarrow D_n}[A(\phi, v) = v'; v \neq v' \text{ and } \phi[v'] = 1] > 1/4.$$  

**Proof.** We start with the definition of $A$. Let $f, g, h$ be the Levin reduction between SAT to GapMCSP[$s_0, s_1$]. Define $A(\phi, v) = h(\phi, \text{iO}(1^{\|\phi(v)\|^{\alpha}}, f(\phi, v))).$ In the following we omit the security parameter $1^{\text{poly}(|\phi(v)|^{\alpha})}$ from the notation.

Next, we show that $A(\phi, v)$ returns $v' \neq v$ that satisfies $\phi$ with probability at least $1/4$. By Claim 3.5, such $v'$ exists with all but negligible probability over a random sample $(\phi, v) \leftarrow D_n$. For the constant $\delta > 0$ from Claim 3.6 let $G$ be the set of all $(\phi, v)$ such that $s_0(\|f(\phi)\|) \geq n^{\delta}$ and that exists $v' \neq v$ with $\phi[v'] = 1$. By Claim 3.6, $\Pr_{(\phi, v) \leftarrow D_n}[\phi \in G] \geq 1 - \text{neg}(n)$. In the following, fix $n \in \mathbb{N}$, and fix $(\phi, v) \in G$, and $v' \neq v$ with $\phi[v'] = 1$.

By the correctness of $f$ and $g$, $g(\phi, v')$ and $g(\phi, v)$ are two circuits with size at most $s_0(\|f(\phi)\|)$ with the same truth table $f(\phi)$. We assume without loss of generality that $|g(\phi, v)| = |g(\phi, v')| = s_0(\|f(\phi)\|)$. By the assumption on the overhead time of the obfuscator iO, we get that the size of the output of $\text{iO}(g(\phi, v))$ and $\text{iO}(g(\phi, v'))$ is at most

$$p(|g(\phi, v)|, |g(\phi, v')|^{\alpha}) = p(s_0(\|f(\phi)\|), (s_0(\|f(\phi)\|))^{\alpha}) < s_1(\|f(\phi)\|).$$  

Thus, the output $\text{iO}(g(\phi, v))$ is a witness that $f(\phi)$ is not a No instance of GapMCSP[$s_0, s_1$], and by the definition of $h$, $h(\phi, \text{iO}(g(\phi, v)))$ returns a witness that $\phi \in \text{SAT}$. Similarly, the same holds for $v'$: $h(\phi, \text{iO}(g(\phi, v'))) \text{ returns a witness that } \phi \in \text{SAT}.$

Lastly, we use the security of iO to claim that $h(\phi, \text{iO}(g(\phi, v))) \neq v$ with a good probability. By the security of the obfuscator, a random sample $(\phi, v)$ and $g(\phi, v)$ computes the same function $f(\phi)$ the output distributions of $\text{iO}(g(\phi, v))$ and $\text{iO}(g(\phi, v'))$ are indistinguishable. Moreover, since the iO is secure against non-uniform algorithms, the above distributions are indistinguishable also given $(\phi, v, v')$ (importantly, the size of $(\phi, v, v')$ is polynomial in the security parameter and in the size of the circuit $g(\phi, v)$ when $s_0(\|f(x)\|) \geq n^{\delta}$). In particular, by data processing, the distributions $h(\phi, \text{iO}(g(x, v)))$ and $h(\phi, \text{iO}(g(x, v'))) \text{}$ must be indistinguishable.

By the definition of $A$, we get that

$$\Pr[A(\phi, v) = v] \leq \Pr[A(\phi, v') = v] + \mu(s_0(\|f(\phi)\|))$$  

for some negligible function $\mu$. Since $(\phi, v) \in G$, for every large enough $n$ we get that

$$\Pr[A(\phi, v) = v] \leq \Pr[A(\phi, v') = v] + \mu(s_0(\|f(\phi)\|)) \leq \Pr[A(\phi, v') \neq v'] + 1/3,$$
which implies that
\[ 1 - \Pr[A(\phi, v) \neq v] \leq \Pr[A(\phi, v') \neq v'] + 1/3, \]
or that
\[ 1/2 \cdot (\Pr[A(\phi, v) \neq v] + \Pr[A(\phi, v') \neq v']) \geq 1/3. \quad (1) \]
To finish the proof, consider the distribution \( \mathcal{D}'_n \), in which we sample \((\phi, v) \leftarrow \mathcal{D}_n\), and then if \((\phi, v) \in \mathcal{G}\), we sample a random \(v' \neq v\) such that \(\phi[v'] = 1\) (or let \(v' = v\) if \((\phi, v) \notin \mathcal{G}\)). We then output \((\phi, v, v')\).

We get that
\[
\Pr_{(\phi,v) \leftarrow \mathcal{D}_n}[A(\phi, v) \neq v] \\
\geq \Pr_{(\phi,v) \leftarrow \mathcal{D}_n}[A(\phi, v) \neq v \mid (\phi, v) \in \mathcal{G}] \cdot \Pr_{(\phi,v) \leftarrow \mathcal{D}_n}[(\phi, v) \in \mathcal{G}] \\
= \Pr_{(\phi,v) \leftarrow \mathcal{D}_n}[A(\phi, v) \neq v \mid (\phi, v) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\
= \Pr_{(\phi,v_0,v_1) \leftarrow \mathcal{D}_n}[A(\phi, v_0) \neq v_0 \mid (\phi, v_0) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\
= \Pr_{(\phi,v_0,v_1) \leftarrow \mathcal{D}_n,b \leftarrow \{0,1\}}[A(\phi, v_b) \neq v_b \mid (\phi, v_b) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\
= 1/2 \cdot \sum_{b \in \{0,1\}} \Pr_{(\phi,v_0,v_1) \leftarrow \mathcal{D}_n}[A(\phi, v_b) \neq v_b \mid (\phi, v_b) \in \mathcal{G}] \cdot (1 - \text{neg}(n)) \\
\geq 1/3 - \text{neg}(n). 
\]
where the third equality holds since the distribution of \((\phi, v_0)\) and \((\phi, v_1)\) are identical for \((\phi, v_0, v_1) \leftarrow \mathcal{D}_n\), and the last inequality by Equation (1).

We are now ready to prove Theorem 3.2.

Proof of Theorem 3.2. Assume that iO and subexponential one-way function exist. By Theorem 2.11, there exists a TCR with security against subexponential adversaries.

Assume there exists Levin reduction from SAT to GapMCSP\([s_0, s_1]\), and let \(\mathcal{D}\) be the distribution defined above. By Claim 3.5, there is no efficient algorithm that given a random sample \((\phi, v)\) from \(\mathcal{D}\) finds \(v' \neq v\) such that \(\phi[v'] = 1\) with non-negligible probability. But by Lemma 3.8, there exists such an algorithm that succeed with probability \(1/4\), which is a contradiction.

\[ \square \]

3.2 Randomized Levin Reductions

In this part we generalize Theorem 3.2 to hold with respect to randomized reductions. We prove the following theorem.

**Theorem 3.9.** Let \(0 \leq \epsilon \leq 1/30\) be a constant, and let \(p: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}\) be a function. Assume that there exist iO with overhead \(p\), and subexponentially-secure one-way function. Then for any constant \(\alpha > 0\) and for any pair of efficiently computable functions \(s_0, s_1: \mathbb{N} \rightarrow \mathbb{N}\) for which \(s_1(n) > p(s_0(n), (s_0(n))^{\alpha})\), it holds that GapMCSP\([s_0(n), s_1(n)]\) is not NP complete with respect to randomized Levin reductions with \(\epsilon\)-error.

Theorem 1.1 (for randomized reductions) directly follows by Theorem 3.9 and the observation that the overhead \(p\) is always bounded by polynomial. Using the result of [AJS17], we get the following corollary.
Corollary 3.10. Let $0 \leq \epsilon \leq 1/30$ be a constant. Assume subexponential-secure iO, and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every constant $\alpha > 0$, and for any efficiently computable function $s_0$, it holds that GapMCSP[$s_0(n),(2+\alpha)s_0(n)$] is not NP complete with respect to randomized Levin reductions with $\epsilon$-error.

The proof of Theorem 3.9 is similar to the proof of Theorem 3.2. Let iO be an indistinguishability obfuscator with overhead $p$, and $T: \{0,1\}^n \rightarrow \{0,1\}^{n-\omega(\log n)}$ be a TCR secure against subexponential adversaries. Let $D = \{D_n\}_{n \in \mathbb{N}}$ be the same distribution as defined in the proof of Theorem 3.2.

The following claim is the analog of Claim 3.6 for randomized reductions.

Claim 3.11. Let $(f,g,h)$ be a randomized Levin reduction with $\epsilon$-error from SAT to GapMCSP[$s_0,s_1$]. Then there exists a constant $\delta > 0$ such that

$$\Pr_{(\phi,v) \leftarrow D_n,r_1 \leftarrow \{0,1\}^*}[s_0(|f(\phi;r_1)|) \geq n^\delta] \geq 1 - 2\epsilon - \text{neg}(n)$$

Proof. The proof follows the same lines as the proof of Claim 3.6. Specifically, let for $\delta > 0$, $A$ be the algorithm described in the proof of Claim 3.6. We will show that

$$\Pr_{x \leftarrow \{0,1\}^n}[A(T(x)) \in T^{-1}(T(x))] \geq \Pr_{(\phi,v) \leftarrow D_n,r_1 \leftarrow \{0,1\}^*}[s_0(|f(\phi)|) < n^\delta] - 2\epsilon.$$ 

The claim will then follow by Claim 2.13.

By the definition of randomized Levin reductions, with probability at least $1 - \epsilon$ over the choice of $r_1$, it holds that $h$ succeed to convert a witness for $f(\phi;r_1)$ to a witness for $\phi$ with probability at least $1 - \epsilon$. By the union bound, with probability at least

$$1 - \Pr_{(\phi,v) \leftarrow D_n,r_1 \leftarrow \{0,1\}^*}[s_0(|f(\phi;r_1)|) < n^\delta] - \epsilon$$

over the choice of $(\phi,v) \leftarrow D_n$ and $r_1$, it holds that both $s_0(|f(\phi;r_1)|) < n^\delta$, and $h$ converts witnesses for $f(\phi;r_1)$ to witnesses for $\phi$ with probability at least $1 - \epsilon$. In this case, $A$ finds a witness for $f(\phi;r_1)$ and outputs a pre-image of $T$ with probability $1 - \epsilon$.

Using the union bound again, we get that $A$ finds such a pre-image with probability at least

$$1 - \Pr_{(\phi,v) \leftarrow D_n,r_1 \leftarrow \{0,1\}^*}[s_0(|f(\phi;r_1)|) \geq n^\delta] - 2\epsilon$$

as claimed. \[\square\]

The next lemma generalized Lemma 3.8, to shows it is possible to use iO and randomized Levin reduction to find collisions in the TCR.

Lemma 3.12. Let iO be indistinguishability obfuscator with overhead $p$, and let $\epsilon,s_0$ and $s_1$ as in Theorem 3.9. Assume that there exists a randomized Levin reduction with $\epsilon$-error from SAT to GapMCSP[$s_0,s_1$]. Then there exists an efficient algorithm $A$ such that for every large enough $n \in \mathbb{N}$

$$\Pr_{(\phi,v) \leftarrow D_n}[A(\phi,v) = v'; v \neq v' \text{ and } \phi[v'] = 1] > 1/4 - 5\epsilon.$$
Proof. We start with the definition of $A$. Let $f, g, h$ be the Levin reduction between SAT to GapMCSP$[s_0, s_1]$, and define $A$ to be the algorithm that on input $\phi, v$, outputs

$$h(\phi, iO(1^{\lfloor g(\phi, v; r_1) \rfloor}), g(\phi, v; r_1); r_1, r_2),$$

for a random choice of randomness $r_1, r_2$ for $g, h$. In the following we omit the security parameter $1^{\lfloor g(\phi, v; r_1) \rfloor}$ from the notation.

Next, we show that $A(\phi, v)$ returns $v' \neq v$ that satisfies $\phi$ with probability at least $1/4$. Let $G$ be the set of all SAT formulas $\phi$ such that there are $v \neq v'$ such that $\phi[v] = \phi[v'] = 1$.

Let $\delta > 0$ be the constant from Claim 3.11. In the following, we say that a randomness $r_1$ is good for a formula $\phi$ and a satisfying assignments $v$, if it holds that (1) $s_0(|f(\phi; r_1)|) \geq n^\delta$, (2) $g(\phi, v; r_1)$ is a circuit of size at most $s_0(|f(\phi; r_1)|)$ that computes $f(\phi; r_1)$, and (3), for any circuit $C$ of size less than $s_1(|f(\phi; r_1)|)$ which computes $f(\phi; r_1)$, it holds that $h(\phi, C; r_1, r_2)$ is a satisfying assignments for $\phi$ with probability at least $1 - \epsilon$ over the choice of $r_2$. That is, $r_1$ is good if the output of $f(\phi; r_1)$ is not too short, and if the reduction succeed in converting witnesses from SAT to GapMCSP using the randomness $r_1$.

By the definition of Levin reductions with $\epsilon$-error a random $r_1$ fulfils the last two requirements with probability at least $1 - \epsilon$. Using Claim 3.11 and the union bound, we get that a random $r_1$ is good for $(\phi, v)$ with probability at least $1 - 3\epsilon - \text{neg}(n)$.

For $\phi \in G$, and two satisfying assignments $v \neq v'$, let $R_{\phi, v, v'}$ be the set of all random strings $r_1$ such that $r_1$ is good both for $(\phi, v)$ and for $(\phi, v')$. Using the union bound again, we get that

$$\Pr_{r_1 \leftarrow \{0, 1\}^n} [r_1 \in R_{\phi, v, v'}] \geq 1 - 6\epsilon - \text{neg}(n).$$

We continue as in the proof of Lemma 3.8. In the following, fix $\phi \in G$ and two satisfying assignments $v \neq v'$, and fix $r_1 \in R_{\phi, v, v'}$.

By the definition of $R_{\phi, v, v'}$, $g(\phi, v; r_1)$ and $g(\phi, v'; r_1)$ are two circuits with size at most $s_0(f(\phi))$ with the same truth table $f(\phi; r_1)$. We assume without loss of generality that $|g(\phi, v)| = |g(\phi, v')| = s_0(|f(\phi)|)$. As in the proof of Lemma 3.8, by the assumption on the overhead of the obfuscator $iO$, we get that the size of the output of $iO(g(\phi, v; r_1))$ and $iO(g(\phi, v; r_1))$ is less than $s_1(|f(\phi; r_1)|)$. Thus, the output $iO(g(\phi, v; r_1))$ is a witness that $f(\phi; r_1)$ is not a No instance of GapMCSP$[s_0, s_1]$, and by the definition of $h$ and $R_{\phi, v, v'}$, $h(\phi, iO(g(\phi, v; r_1)))$ returns a witness that $\phi \in \text{SAT}$ with probability at least $1 - \epsilon$ over the choice of $r_2$. Similarly, the same holds for $v'$: $h(\phi, iO(g(\phi, v')))$ returns a witness that $\phi \in \text{SAT}$ with the same probability.

Lastly, we use the security of $iO$ to claim that $h(\phi, iO(g(\phi, v; r_1); r_1, r_2))$ outputs an satisfying assignment to $\phi$ which is not equal to $v$ with a good probability. By the security of the obfuscator, and since $g(\phi, v; r_1)$ and $g(\phi, v'; r_1)$ computes the same function $f(\phi; r_1)$ the output distributions of $iO(g(\phi, v; r_1))$ and $iO(g(\phi, v'; r_1))$ are indistinguishable. Moreover, by the non-uniform security, the above distributions are indistinguishable also given $(x, v, v', r_1)$. In particular, by data processing, the distributions $h(\phi, iO(g(x, v; r_1)); r_1, r_2)$ and $h(\phi, iO(g(x, v'; r_1)); r_1, r_2)$ must be indistinguishable. Let $A(\phi, v; r_1)$ be the output of $A(\phi, v)$ when we fix the randomness $A$ uses for $f$ to be $r_1$. In the following we assume without loss of generality that whenever $A$ do not output a satisfying assignment for $\phi$, it outputs $\perp$. By the definition of $A$, when $r_1 \in R_{\phi, v, v'}$ we get that

$$\Pr[A(\phi, v; r_1) = v] \leq \Pr[A(\phi, v'; r_1) = v] + \mu(s_0(|f(\phi)|))$$

for some negligible function $\mu$. As in the proof of Lemma 3.8, this implies that

$$1/2 \cdot (\Pr[A(\phi, v; r_1) \neq v] + \Pr[A(\phi, v'; r_1) \neq v]) \geq 1/3.$$  

(3)
Since $h$ fails with probability at most $\epsilon$, we get that
\begin{equation}
1/2 \cdot (\Pr[A(\phi, v; r_1) \notin \{v, \perp\}] + \Pr[A(\phi, v'; r_1) \notin \{v', \perp\}]) \geq 1/3 - \epsilon.
\end{equation}

To finish the proof, consider the distribution $D'_n$, in which we sample $(\phi, v) \leftarrow D_n$, and then if $\phi \in \mathcal{G}$, we sample a random $v' \neq v$ such that $\phi[v'] = 1$ (otherwise we let $v' = v$). We then output $(\phi, v, v')$.

We get that
\begin{align*}
\Pr_{(\phi, v) \leftarrow D_n, r_1 \leftarrow \{0,1\}^*}[A(\phi, v; r_1) \notin \{v, \perp\}] \\
= \Pr_{(\phi, v_0, v_1) \leftarrow D'_n, r_1 \leftarrow \{0,1\}^*}[A(\phi, v_0; r_1) \notin \{v_0, \perp\}] \\
\geq \Pr_{(\phi, v_0, v_1) \leftarrow D'_n}[A(\phi, v_0; r_1) \notin \{v_0, \perp\} | \phi \in \mathcal{G}, r_1 \in R_{\phi,v_0,v_1}] \cdot \Pr[r_1 \in R_{\phi,v_0,v_1} | \phi \in \mathcal{G}] \cdot \Pr[\phi \in \mathcal{G}] \\
\geq \Pr_{(\phi, v_0, v_1) \leftarrow D'_n}[A(\phi, v_0; r_1) \notin \{v_0, \perp\} | \phi \in \mathcal{G}, r_1 \in R_{\phi,v_0,v_1}] \cdot (1 - 6\epsilon - \neg(n))(1 - \neg(n)) \\
\geq (1/3 - \epsilon) \cdot (1 - 6\epsilon - \neg(n))(1 - \neg(n)) \\
\geq 1/4 - 7\epsilon.
\end{align*}

where the second inequality holds by Equation (4) and by Claim 3.11, the third equality holds since the distribution of $(\phi, v_0)$ and $(\phi, v_1)$ are identical for $(\phi, v_0, v_1) \leftarrow D'_n$, by a similar argument as in the proof of Lemma 3.8, and the last inequality holds for large enough $n$ and for a small enough constant $\epsilon$.

We are now ready to prove Theorem 3.2.

**Proof of Theorem 3.2.** Assume that IO and subexponentially-secure one-way function exist. By Theorem 2.11, there exists a TCR with security against subexponential adversaries.

Assume there exists Levin reduction from SAT to GapMCSP[$s_0, s_1$], and let $D$ be the distribution defined above. By Claim 3.5, there is no efficient algorithm that given a random sample $(\phi, v)$ from $D_n$ finds $v' \neq v$ such that $\phi[v'] = 1$ with non-negligible probability. But by Lemma 3.12, there exists such an algorithm that succeed with probability $1/4 - 7\epsilon$, which is a contradiction when $\epsilon < 1/28$.

### 4 GapMKPK is not NP-complete under Levin Reductions

In this section we prove our result for MKPK. That is, we prove that (under cryptographic assumptions) there is no Levin reduction from SAT to the following promise problem. For $p, t \in \text{poly}$, let GapMKPK[$s_0, s_1$] be the following promise problem:

- $\mathcal{Y} = \{x \in \{0,1\}^n : K^{t(n)}(x) \leq s_0(n)\}$
- $\mathcal{N} = \{x \in \{0,1\}^n : K^{p(t(n))}(x) > s_1(n)\}$
We define the relations $R_Y$ and $R_N$ for Gap$_p$MK$^1$P$[s_0, s_1]$ in the natural way:

$$R_Y = \{(x, P) : P \text{ is a program of length at most } s_0(n) \text{ such that } U(P, 1^{t(|x|)}) = x\},$$

and,

$$R_N = \{(x, P) : P \text{ is a program of length at most } s_1(n) \text{ such that } U(P, 1^{p(t(|x|))}) = x\}.$$

The proof follows the same line as the proof of Theorem 3.2, where we replace the iO with randomized encoding for Turing machines with indistinguishability-based security [AJ15].

**Definition 4.1** (Randomized encoding for TM). A pair of efficient randomized algorithms $(Enc, Dec)$ is randomized encoding for TMs if the following holds: Let $M$ be a TM and $x \in \{0, 1\}^*$ be an input, $\lambda \in \mathbb{N}$ be a security parameter and let $T \in \mathbb{N}$ be a bound on the running time of $M(x)$. Then

1. (Correctness:) $\Pr[Dec(Enc(1^\lambda, M, x, T)) = M(x)] = 1$

2. (Efficiency:) $Enc(1^\lambda, M, x, T)$ runs in time $\text{poly}(\lambda, |M|, |x|, \log T)$ and $Dec(M(x))$ runs in time $\text{poly}(\lambda, |M|, |x|, t)$ for $M(x) \leftarrow Enc(1^\lambda, M, x, T)$ and where $t \leq T$ is the running time of $M(x)$, and,

3. (Security:) For every ppt algorithm $A$ and every $s \in \text{poly}$ there exists a negligible function $\mu$, such that for every TM $M$ and two inputs $x_0, x_1$ such that $M(x_0) = M(x_1)$, $|M| \leq s(\lambda), |x_0| \leq s(\lambda), |x_1| \leq s(\lambda)$ and the running time of $M$ on $x_0$ at most $s(\lambda)$ and is the same as the running time of $M$ on $x_1$, the following holds:

$$\left| \Pr[A(Enc(1^\lambda, M, x_0, T)) = 1] - \Pr[A(Enc(1^\lambda, M, x_1, T)) = 1] \right| = \mu(\lambda).$$

We say that $(Enc, Dec)$ has overhead $p$ if $|Enc(1^\lambda, M, x, T)| \leq p(|M|, |x|, T, \lambda)$ with probability 1.

Using randomized encoding, we get the following theorem.

**Theorem 4.2.** Let $0 \leq \epsilon \leq 1/30$ be a constant. Assume that randomized encoding for TMs with overhead $q$, and subexponentially-secure one-way function exist. Then there exists a constant $c \in \mathbb{N}$ such that for every constant $\alpha > 0$, for any $t \in \text{poly}$ and any efficiently computable functions $s_0, s_1 : \mathbb{N} \rightarrow \mathbb{N}$ for which

$$s_1(n) > q(c, s_0(n) + c \log(t(n)) + c \log(s_0(n)), \log t(n), (s_0(n))^\alpha),$$

and for every large enough polynomial $p$, it holds that Gap$_p$MK$^1$P$[s_0, s_1]$ is not NP complete with respect to randomized Levin reductions with $\epsilon$-error.

By the results of [LPST15; KLW15] such randomized encoding with polynomial overhead $q$ for poly-time TMs can be constructed assuming one-way functions, subexponentially-secure iO for circuits and injective PRG (that can be constructed from one-way permutation). Together with Theorem 4.2 we get Theorem 1.3. As in Theorem 3.2, we can relax the requirement for subexponentially-secure one-way function if we only want to exclude honest reductions.

[AJS17] constructed iO for TM with multiplicative overhead. By combining the construction of randomized encoding for TMs of [LPST15] with the iO of [AJS17], we get randomized encoding with multiplicative overhead.
**Theorem 4.3.** Assuming subexponentially-secure iO and subexponentially secure re-randomizable encryption schemes, there exists a randomized encoding for TMs scheme with overhead $q(|M|, |x|, T, \lambda) = 2(|M| + |x|) + \text{poly}(\lambda, \log T)$.

We get the following corollary.

**Corollary 4.4.** Let $0 \leq \epsilon \leq 1/3$ be a constant. Assume subexponential-secure iO, and subexponentially-secure one-way function exist and assume subexponential DDH or LWE. Then for every constant $\alpha > 0$, and for any efficiently computable function $s_0$, it holds that Gap$_p$MK$^1$P$s_0(n), (2 + \alpha)s_0(n)$ is not NP complete with respect to randomized Levin reductions with $\epsilon$-error.

**Proof of Theorem 4.2.** For easy of notation, we explain how to modify the proof of Theorem 3.2 to get the proof of Theorem 4.2 for deterministic reductions. Similar changes to the proof of Theorem 3.9 yield the result for randomized reductions.

We only need to change the proof of Lemma 3.8. Let $(f, g, h)$ be the Levin reduction from SAT to Gap$_p$MK$^1$P$s_0, s_1$, and assume that for every $(\phi, v)$ in the support of $D$, $g(\phi, v)$ output a program of length exactly $s_0(|f(\phi)|)$ that runs in time exactly $t(|f(\phi)|)$ (this can be assume by adding $O(\log t(n) + \log s_0(n))$ bits to the description of $g(\phi, v)$). Let $U$ be a universal TM and $(Enc, Dec)$ be randomized encoding for TMs. Consider the algorithm

$$A(\phi, v) = h(\phi, g(\phi, v)),$$

where $g(\phi, v)$ is a program that runs $Dec$ on $\hat{P}$ for $\hat{P} \leftarrow Enc(1|g(\phi, v)|^{\alpha}, U, g(\phi, v), t(|f(\phi)|))$. That is, we replace the the iO in the construction of $A$ from the proof of Lemma 3.8, with a randomized encoding of $U(g(\phi, v))$. Since for every two witnesses $v, v'$ of $\phi$ it holds that $U(g(\phi, v)) = U(g(\phi, v')) = f(\phi)$, we get that $g(\phi, v)$ and $g(\phi, v')$ are indistinguishable.

By the overhead of the randomized encoding scheme,

$$|g(\phi, v')| \leq q(|U|, s_0(n) + O(\log t(n)) + \log(s_0(n)), \log t(n), |g(\phi, v)|^{\alpha}).$$

By the efficiency of $Dec$, the running time of $g(\phi, v')$ is at most $\text{poly}(s_0(|f(\phi)|), t(|f(\phi)|)) = \text{poly}(t(|f(\phi)|))$, where the equality holds since $s_0(|f(\phi)|) \leq |f(\phi)| + O(1)$ or the Gap$_p$MK$^1$P$s_0, s_1$ problem is trivial. Thus, by taking $p$ be a polynomial that bound the running time of $g(\phi, v')$, we get that $g(\phi, v')$ is a witness that $f(\phi)$ is not a No instance. The proof continues along the same lines as the proof of Lemma 3.8.

**References**


Ilango [Ila23] show a reduction from τ-Frequency Set Cover to both GapMCSP and Gap_p MK^1P with a random oracle O. Here we explain why this reduction is a Levin reduction.

Given a witness to the τ-Frequency Set Cover, the construction of a witness for the output of the reduction form Gap_p MK^1P is straightforward, and the construction of the witness for GapMCSP uses the construction of [Lup58] that can be made efficient (recall that the running time can be polynomial in the truth-table of the circuit).

We briefly explain how the proof in [Ila23] implies that given a witness for the GapMCSP or Gap_p MK^1P instances that we get from the reduction, we can reconstruct a witness for the τ-Frequency Set Cover instance. Specifically, this can be done by considering the set of queries made by the GapMCSP or Gap_p MK^1P witnesses to the random oracle O. For concreteness, we focus on the reduction for GapMCSP (the proof for Gap_p MK^1P is of the same lines).

We start with a short description of the reduction. Given an instance \( \phi = (S_1, \ldots, S_m \subseteq [n]) \), recall that we want to find a small subset of \( J \subseteq [m] \) such that \( \cup_{j \in J} S_j = [n] \). The reduction samples for each such set \( S_j \) a secret key \( sk_j \), and for every element in \( i \in [n] \) a random value \( v_i \). It then finds for each \( i \) and for each \( j \) such that \( i \in S_j \), a value \( c_{i,j} \) such that \( O(i, sk_j, c_{i,j}) = v_i \). Then the truth table that the reduction outputs is the concatenation of \( c_{i,j} \) and \( v_i \) for all \( i \in [n], j \in [m] \).

The hope is that any circuit that computes this truth table will have the values of \( sk_j \) hardcoded to it for every \( j \) in the minimal cover \( J \). While this is not true, we explain below that (with high probability over the oracle \( O \) we can extract an approximation of \( J \) using the oracle calls the circuit makes to \( O \). Specifically, for the gap problem used in [Ila23], we need to find a set cover of size smaller than \( n/3 \).

Let \( \phi \) be a τ-Frequency Set Cover, and let \( x = f(\phi) \) be the output of the reduction. [Ila23] shows that when \( \phi \) is a No instance, the probability over the choice of \( O \) that any fixed algorithm that makes bounded number of queries to \( O \) can output \( x \) is exponentially small in the length of
Then, by the union bound over all possible small circuits (or program), [Ila23] shows that no such circuit that outputs \( x \) exists (with high probability over \( O \)). We observe that with the same exponentially small probability, if an algorithm can output \( x \), then we can extract from it a set cover of size smaller than \( n/3 \). By the same union bound over all circuits, we get that we can extract such a solution from all of the small circuits that output \( x \).

The way the probability of a algorithm \( A \) to output \( x \) is bounded in [Ila23] by considering the set \( skHit \) of all the indexes \( j \in [m] \) such that \( A \) queried \( O \) on \((i, sk_j, c)\) for some \( i \) and \( c \). Then, let \( Missed = [n] \setminus \bigcup_{j \in skHit} S_j \). Now, if the total size of \( skHit \) and \( Missed \) is less than \( n/3 \), we can take \( skHit \) together with some trivial cover of \( Missed \) as our set cover, and we are done (impotently, \( skHit \) and \( Missed \) can be computed from the algorithm). We thus left to show that for any algorithm \( A \), the probability that \( A \) outputs \( x \) and \(|skHit| + |Missed| \geq n/3\) is exponentially small.

This follows by the proof in [Ila23]: In the proof of Proposition 37, we can just remove from the first sum terms with \(|skHit| + |Missed| \leq n/3\). Note that by the the information revealed by the third step in the proof, we can compute the sets \( skHit \) and \( Missed \), and thus we can check if \(|skHit| + |Missed| < n/3\) without revealing any new information.