DIMACS Final Report: Equivalence between Restricted Non-interactive Statistical Zero-knowledge Classes

Jacob Gray, Saachi Mutreja, Pengxiang Wang
With Eric Allender and Harsha Tirumala

July 29th, 2022

Abstract

We investigate the class $\mathsf{NISZK_L}$ and its connections to other classes, mainly in showing equivalences between similarly-defined subclasses of $\mathsf{NISZK_L}$. This class was originally introduced in [All+21] to gain insight on the hardness of MCSP through the hardness of MKTP. There, it was shown that MKTP is hard for $\overline{\mathsf{NISZK_L}}$, which makes this class of interest in efforts to place MKTP and MCSP in a more solid position in the complexity landscape.

We start by expanding upon results from the previous summer's REU project with Allender, which showed that $\mathsf{NISZK_L} = \mathsf{NISZK_{AC^0[\oplus]}}$. We show that the parity requirement can be removed, and $\mathsf{NISZK_L} = \mathsf{NISZK_{AC^0}}$. We then prove a new result expanding in the other direction, towards proving a conjecture relating to a stronger NISZK subclass. This conjecture is that $\mathsf{NISZK_L} = \mathsf{NISZK_{DET}}$; our result shows that $\mathsf{NISZK_L} = \mathsf{NISZK_{PM}}$, where PM is the class of problems reducible to perfect matching. We then show that for sufficiently strong classes $\mathsf{A} \subseteq \mathsf{B} \subseteq \mathsf{NISZK_A}$, we have $\mathsf{NISZK_A} = \mathsf{NISZK_{A,B}}$, which has several implications and provides a better intuition for these classes when varying the powers of verifier and simulator. We end by proving a closure result for $\mathsf{NISZK_L}$, and mention other attempted closure results.

1 Introduction

This project mainly revolves around the recently introduced class $\mathsf{NISZK_L}$ (non-interactive, statistical zero knowledge proof systems with log-space verifiers and simulators) and related results. To introduce this class, we start with the notion of proof systems, which are computational models involving a "weak verifier" (protypically a polynomial-time Turing machine) and a computationally unbounded prover. The prover and verifier exchange messages in an attempt to convince the verifier of some property of an input string. This can be modified to a "zero knowledge" protocol, where we add an additional simulator (usually of the same power as the verifier) which must emulate the transcripts output by the prover. The intuition for this is that if the simulator can produce the prover's transcripts itself, no additional information beyond the goal of the proof is given to the verifier, since the verifier could always have derived the prover's responses itself. The set of problems computable by this variant of proof system is called SZK if the simulator must have a statistically similar distribution to the prover, and both the verifier and simulator are polynomial-time Turing machines. To get NISZK, we add non-interaction, which simply means that the prover only sends a single message to the verifier. This proof system is also augmented with a shared uniformly random

string, called the reference string, which can be seen as "shared randomness" in place of verifier communication.

We also work with the classes $\mathcal{PREN}$ and $\mathcal{SREN}$, introduced in [AIK06], which are functions with randomized encodings in NC^0. These are useful in that weaker classes like L can compute/use these randomized encodings of functions from more powerful classes (such as NL). In particular, they were used in [All+21] to show that EA_{NC^0} was hard for $\mathsf{NISZK}_\mathsf{L}$. For that reason they seem especially useful, since other classes that admit NC^0 randomized encodings seem likely to also be equivalent to $\mathsf{NISZK}_\mathsf{L}$.

2 Classes equal to $\mathsf{NISZK}_\mathsf{L}$

Theorem 2.1. $\mathsf{NISZK}_{\mathsf{AC}^0} = \mathsf{NISZK}_\mathsf{L}$

The proof uses the same idea as in [GSV99] to show $EA \in \mathsf{SZK}$ and [All+21] to show that $EA_{\mathsf{NC}^0} \in \mathsf{NISZK}_\mathsf{L}$. This method involves concatenating many copies of the input circuit, hashing this new circuit, taking many copies of this hashed circuit, and then hashing this new circuit again to produce a circuit whose distribution is either very close to uniform or has small support. Since the proof deals with AC^0, the majority of the work in adapting this method is in showing that the hashing steps can be done in AC^0, since the other proofs did not use AC^0 hashing functions.

Theorem 2.2. $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{NL}$

The first attempted approach to this problem was to try analyzing the entropy of an $\mathcal{SREN}$ encoding of an NL simulator for some promise problem Π. Since EA_{NC^0} is complete for $\mathsf{NISZK}_\mathsf{L}$, it seemed plausible that entropy analysis of this NC^0 circuit could determine whether the encoding was one for an input in Π_{YES} or for one in Π_{NO}. However, this approach faced an issue because a NO instance could still have a high entropy despite being over a small support, and then entropy alone would not be enough to distinguish between YES and NO instances.

Much later in the project, a new approach arose to proving $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{NL}$: this involved a result by [RA97] which showed that $\mathsf{UL}/\operatorname{poly} = \mathsf{NL}/poly$. From there, we could use the same approach as from [All+21], where YES or NO instances are distinguished on the basis of the entropy of the NC^0 encoding.

A very similar approach can also be used to prove an analogous result for PM, the class of problems computable by reduction to perfect matching.

Theorem 2.3. $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{PM}$

These results seem to give more confidence in the conjecture that $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{DET}$ from last year's REU project, which we attempted to prove during our project. Our approach was to start by analyzing NISZK in the case where the simulator is in DET and the verifier in L. However, after proving theorem 3.1 and its corollaries, the potential of using this method to prove $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{DET}$ seemed less likely. Despite this roadblock, we re-affirm the conjecture from last summer that these equivalences can be carried up to DET and possibly even further beyond, given our success with classes closer to DET.

Conjecture 2.4. $\mathsf{NISZK}_\mathsf{L} = \mathsf{NISZK}_\mathsf{DET}$

Theorem 2.5. $\mathsf{PM} \in \mathcal{SREN}$

The proof of this theorem is similar to that of theorem 2.3.

Conjecture 2.6. *If $C \subseteq \mathcal{SRE}\mathcal{N}$, then* $\mathsf{NISZK}_C = \mathsf{NISZK}_\mathsf{L}$ *under some weak restriction on the class C (such as some sort of closure properties).*

3 Comparing simulator and verifier power

Prior to our project, there seemed to be no pre-existing literature investigating SZK or NISZK subclasses with verifiers and simulators in different classes. It's trivial that if $\mathsf{NISZK}_\mathsf{X} = \mathsf{NISZK}_\mathsf{Y}$, then any NISZK subclass with verifier and simulator between X and Y will also be the same, but very little intuition was possessed for what would happen otherwise. To form this intuition, a question was posed as to whether $\mathsf{NISZK}_{\mathsf{A,B}} = \mathsf{NISZK}_{\mathsf{B,A}}$, where $\mathsf{NISZK}_{\mathsf{A,B}}$ has an A simulator and B verifier. Informally, the question is whether the verifier or simulator is more important to the proof system, or perhaps whether they are essentially equivalent in terms of importance.

Our investigations seem to suggest that the simulator is more important and harder to make up for, at least in the case where the classes are sufficiently close in power. There are two complementary results:

Theorem 3.1. *If* $\mathsf{A} \subseteq \mathsf{B} \subseteq \mathsf{NISZK}_\mathsf{A}$*, then* $\mathsf{NISZK}_\mathsf{A} = \mathsf{NISZK}_{\mathsf{A,B}}$.

Proof idea: the basic idea of this proof is to use the $\mathsf{NISZK}_\mathsf{A}$ proof system to prove whether the B verifier accepts on a randomly generated proof. The prover will output the same thing as the $\mathsf{NISZK}_{\mathsf{A,B}}$ system's prover, as well as an additional random string for use in proving whether the B verifier accepts. Notice that the prover in $\mathsf{NISZK}_\mathsf{A}$ can already do whatever the prover in $\mathsf{NISZK}_{\mathsf{A,B}}$ does, since the simulators are of the same power, so it also just needs to do this extra step. This should seem to be true for most classes, but it may not hold for classes like NC^0.

Completeness, soundness, and the simulator's distribution being statistically close to the prover's follow naturally from the corresponding completeness, soundness, and simulator's statistical distance from the prover in the original $\mathsf{NISZK}_{A,B}$ proof system. □

Corollary 3.2. *If* $A \subseteq B \subseteq \mathsf{NISZK}_A$*, then* $\mathsf{NISZK}_{B,A} = \mathsf{NISZK}_B$.

This result follows naturally by a similar proof as theorem 3.1.

Corollary 3.3. *If* $\mathsf{A} \subseteq B \subseteq \mathsf{NISZK}_A$*, then* $\mathsf{NISZK}_{A,B} \subseteq \mathsf{NISZK}_{\mathsf{B,A}}$.

This result follows easily from $\mathsf{NISZK}_\mathsf{A} \subseteq \mathsf{NISZK}_\mathsf{B}$ and substitution by theorem 3.1 and corollary 3.2. Notice that if $\mathsf{NISZK}_{\mathsf{A,B}} \supseteq \mathsf{NISZK}_{\mathsf{B,A}}$, then by the above corollary, $\mathsf{NISZK}_\mathsf{A} = \mathsf{NISZK}_\mathsf{B}$ conditioned on $\mathsf{A} \subseteq \mathsf{B} \subseteq \mathsf{NISZK}_\mathsf{A}$.

4 Closure properties

We also attempted to investigate the closure properties of 3 main classes involved in the project: those of $\mathcal{SRE}\mathcal{N}$, SZK_L, and $\mathsf{NISZK}_\mathsf{L}$.

$\mathcal{SRE}\mathcal{N}$: we wanted to know if $\mathcal{SRE}\mathcal{N}$ was closed under MAJORITY, since this being the case seemed promising in proving that larger classes would be in $\mathcal{SRE}\mathcal{N}$. This could be further used in regards to $\mathsf{NISZK}_\mathsf{L}$ equivalences or perhaps OWF results. However, it seemed very difficult to prove anything about this, although it still may very well be the case that this property holds.

Conjecture 4.1. $\mathsf{SZK_L}$ *is closed under* $\leq_T^L$ *reductions.*

We know that $\leq_T^L$ reductions are the same as $\leq_{tt}^L$ reductions from [LL76], so we can focus on just truth table reductions for this question. $\mathsf{SZK_L}$ is already known to be closed under $\leq_{tt}^{\mathsf{NC}^1}$ reductions from [SV03] and [Dvi+10], so this may prove helpful in approaching the case where the reduction is computed by an L machine.

In addition to asking questions about $\mathsf{SZK_L}$, we were also interested in knowing about properties of $\mathsf{NISZK_L}$. Note that by [All+21], MKTP is hard for $\overline{\mathsf{NISZK_L}}$, so $\mathsf{NISZK_L}$ being closed under complementation is likely difficult to prove, even if it is actually true.

Theorem 4.2. $\mathsf{NISZK_L}$ *is closed under* $\leq_{ctt}^{\mathsf{L}}$ *reductions.*

This result uses a simple proof: run each query one after another, and reject if one of them does.

Proof idea: assume there is some log-space machine M computing the queries for language $\Pi \in \mathsf{SZK_L}$ on input x. We can construct a new verifier for the proof system, V, which runs M and checks each query is in Π_{YES} as they are generated. This can be done using the proof system for Π, having V use the same protocol the verifier from Π uses, and having the prover use the same method as well. If at any point a query is returned as not being in Π_{YES}, we have V reject, and otherwise, if it reaches the end of M's computation and all of the instances have been in Π_{YES}, V accepts.

Informally, this is zero-knowledge because this protocol only needs the verifier to send some polynomial number of independent responses for the proof system of Π, and the simulator for our new protocol can use the simulator for the old protocol, running it multiple times on the different instances. Because we only do this polynomial times, the statistical distance between the prover and simulator will still be sufficiently small to fulfill the zero-knowledge conditions.

Conjecture 4.3. $\mathsf{NISZK_L}$ *is closed under* $\leq_{dtt}^{\mathsf{L}}$ *reductions.*

This would follow from the same proof given for $\leq_{ctt}$ reductions if $\mathsf{NISZK_L}$ was known to be closed under complementation, but considering that this would imply that MKTP is hard for $\mathsf{NISZK_L}$ from [All+21], this route seems difficult. However, similar approaches used to show that $\mathsf{SZK_L}$ is closed under $\leq_{dtt}$ reductions may be useful in proving this conjecture.

5 Conclusion

We think the most important results of this project are the expanded $\mathsf{NISZK_L}$ equivalences, with these results expanding the previously known $\mathsf{NISZK}_{\mathsf{AC}^0[\oplus]} = \mathsf{NISZK_L}$ equivalence to $\mathsf{NISZK}_{\mathsf{AC}^0} = \mathsf{NISZK_L} = \mathsf{NISZK}_{\mathsf{PM}}$. The results on swapping verifier and simulator powers seem helpful in gaining intuition for these classes, but it is hard to tell if they will become useful as proof tools. We end in noting that it seems like there is still much room for improvement on our results, both in further developing $\mathsf{NISZK_L}$ and in potentially putting more classes into $\mathcal{PREN}$ and $\mathcal{SREN}$.

6 Acknowledgements

As REU students, we would like to thank our advisor, Eric Allender, his graduate student, Harsha Tirumala for their helpful advice, guidance, and ideas during the program. We would like to thank the DIMACS REU 2022 program and those who helped organize it as well (especially Larry,

Lazaros, and Dawn,) which this research is being conducted as part of and which organized funding for us to be at Rutgers in person during the summer.

We would also like to thank the NSF, with this research being supported by NSF grant CCF-185221.

7 Bibliography

References

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. "Cryptography in NC^0". In: *SIAM Journal on Computing* 36.4 (2006), pp. 845–888. DOI: 10.1137/S0097539705446950. eprint: https://doi.org/10.1137/S0097539705446950. URL: https://doi.org/10.1137/S0097539705446950.

[All+21] Eric Allender et al. "Cryptographic Hardness Under Projections for Time-Bounded Kolmogorov Complexity". In: LIPIcs 212 (2021). Ed. by Hee-Kap Ahn and Kunihiko Sadakane, 54:1–54:17. DOI: 10.4230/LIPIcs.ISAAC.2021.54.

[Dvi+10] Zeev Dvir et al. *On Approximating the Entropy of Polynomial Mappings*. 2010.

[GSV99] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. "Can Statistical Zero Knowledge Be Made Non-interactive? or On the Relationship of SZK and NISZK". In: *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 467–484. DOI: 10.1007/3-540-48405-1_30.

[LL76] Richard Ladner and Nancy Lynch. "Relativization of questions about log space computability." In: 10 (1976), pp. 19–32. DOI: https://doi.org/10.1007/BF01683260.

[RA97] K. Reinhardt and E. Allender. "Making nondeterminism unambiguous". In: *Proceedings 38th Annual Symposium on Foundations of Computer Science*. 1997, pp. 244–253. DOI: 10.1109/SFCS.1997.646113.

[SV03] Amit Sahai and Salil Vadhan. "A Complete Problem for Statistical Zero Knowledge". In: *J. ACM* 50.2 (Mar. 2003), pp. 196–249. ISSN: 0004-5411. DOI: 10.1145/636865.636868. URL: https://doi.org/10.1145/636865.636868.