Iterated Inhomogeneous Polynomials

Jiaxin Guan[⋆] and Mark Zhandry[⋆⋆]

Princeton University & NTT Research, USA

Abstract. Let p be a polynomial mod N, and let $p^{(i)}(x)$ be the result of iterating the polynomial i times, starting at x. In the case where $p(x) = x^2$ and N is the product of two large primes, $p^{(i)}$ both has a natural group structure, and also appears hard to compute in time less than i. This gives rise to interesting cryptographic applications such as time-lock puzzles and verifiable delay functions.
In this work, we consider $p(x) = 2x^2 + 3x + 1$ and $N = 2^n$. Our initial hope was that certain features of this polynomial could be likewise used to build interesting cryptographic applications. To the contrary, and perhaps surprisingly, we show that $p^{(i)}(x)$ can be computed in time logarithmic in i. Moreover, we show that the discrete log problem—finding i given x and $p^{(i)}(x)$—can also be computed efficiently. We conclude with some interesting future directions to explore.

1 Introduction

Polynomials are ubiquitous throughout cryptography, from public key encryption such as RSA [RSA78] to secret sharing [Sha79] to multiparty computation [Yao86, GMW87, BGW88, CCD88]. At the same time, iteration is also ubiquitous in cryptography, from block cipher designs to memory-hard functions [Per09, AS15] such as Scrypt [Per09] to proofs of sequential work [MMV13, CP18, AKK$^+$19, DLM19]. Time-lock puzzles [RSW96] and verifiable delay functions (VDFs) [BBBF18, Wes19, Pie19] even combine both, iterating the homogeneous polynomial x^2. We can also view discrete exponentiation as iterating the polynomial x^2, leading to public key encryption.

In this work, we explore iterating *inhomogeneous* polynomials. Namely, let p be a polynomial mod N for some integer n, and let $p^{(0)}(x) = x, p^{(i)}(x) = p(p^{(i-1)}(x))$ be the result of iterating p for i times to x.

Wishlist. For the iterative application of a polynomial to be useful for cryptography, we may wish for some of the following features:

1. $p^{(i)}(x)$ should have a very large period in i, so that $p^{(i)}(x) \neq x$ for small i.
2. For applications like VDFs, we would like $p^{(i)}(x)$ to be hard to "shortcut"; that is, it should take time roughly i to compute $p^{(i)}(x)$.

[⋆] `jiaxin@guan.io`

[⋆⋆] `mzhandry@cs.princeton.edu`

3. If we want *post-quantum* security, the polynomial p should not correspond to squaring in a natural group law, since iterated squaring in a group can be shortcutted. This requires knowing the group order, which can be easily computed post-quantumly [Sho94].
4. At the same time, group-based VDFs exploit the group structure for verification. So in the absence of a group structure, we would like some algebraic structure that can be used to efficiently verify.
5. Finally, if it is possible to shortcut $p^{(i)}(x)$, then we may try to actually get useful cryptography from this shortcutting. For example, we could consider the following Diffie-Hellman-like protocol, where Alice and Bob choose random i, j, respectively, and broadcast $a = p^{(i)}(0), b = p^{(j)}(0)$. Then they compute the shared secret key as $K = p^{(i+j)}(0) = p^{(i)}(b) = p^{(j)}(a)$. More generally, if we can shortcut $p^{(i)}$, we may hope that we achieve a group *action*, with i acting on x [Cou06, RS06]. In this case, we want discrete logarithms to be hard: given $x, p^{(i)}(x)$, it should be hard to compute i.
   Note that group actions are interesting, since unlike groups, group actions obtain plausible post-quantum security.

Our Setting. We start with the following observation:

Theorem 1. *$2x^2 + 3x + 1$ is a permutation on $\mathbb{Z}_{2^n}$, and this permutation is a cycle of length 2^n.*

Thus, setting $p(x) = 2x^2 + 3x + 1$ and $N = 2^n$, we satisfy Wishlist Item 1. We also show that $p(x)$ does not correspond to squaring in any natural group, satisfying Wishlist Item 3. Since p does not correspond to a group law, that the order of p is known does not immediately invalidate post-quantum security.

Our initial hope, based on these observations, were that shortcutting (Wishlist Item 2) is hard, while there is still some structure to exploit for applications (Wishlist Item 4). Or alternatively, we hoped at least that discrete logarithms were hard (Wishlist Item 5).

2 Fail 1: Efficient Short-cutting

Unfortunately, we show how to shortcut the computation of $p^{(i)}$, computing it in time $\mathsf{poly}(n)$, ruling out Wishlist Item 2:

Theorem 2. *There exists a deterministic algorithm running in time polynomial in n, which computes $p^{(i)}(x)$ for any $i, x \in \mathbb{Z}_{2^n}$.*

Note that by Theorem 1, we only need to consider i in $\mathbb{Z}_{2^n}$, since any two i's differing by a multiple of 2^n give the same result.

To prove the theorem, we define the set $\mathcal{Q}_n$ of polynomials q over $\mathbb{Z}_{2^n}$ of the following form: $q(x) = a_0 + a_1 x + 2a_2 x^2 + 4a_3 x^3 + \cdots + a_j 2^{j-1} x^j + \ldots$.

We then make two observations: first, all a_j for $j > n$ can be taken to be 0, since those terms will have been multiplied by a factor of 2^n [1]. Second,

[1] With a little more care, we can also eliminate all a_j for $j \gtrsim n/2$.

composing two polynomials in $\mathcal{Q}_n$ gives another polynomial in $\mathcal{Q}_n$. Hence, $\mathcal{Q}_n$ forms a ring under polynomial addition and composition.

Importantly, the above means that $p^{(i)} \in \mathcal{Q}_n$, and can be represented as a polynomial of degree at most n. Moreover, we can write $p^{(i)}(x) = p^{(i-1)}(p(x))$, which allows us to write the coefficients for $p^{(i)}$ as linear combinations of the coefficients of $p^{(i-1)}$. Computing $p^{(i)}$ then becomes equivalent to taking a matrix power M^i, where M is the matrix corresponding to the linear combinations. We can then compute $p^{(i)}(x)$ by simply evaluating the polynomial.

With Wishlist Item 2 ruled out, we can at least hope for discrete logarithms (Wishlist Item 5) to be hard, giving rise to plausible post-quantum Diffie-Hellman protocols.

3 Fail 2: Efficient Discrete Logs

Unfortunately, discrete logarithms can actually be efficiently computed:

Theorem 3. *There exists a randomized algorithm running in time polynomial in n, which computes i given $x, y = p^{(i)}(x) \in \mathbb{Z}_{2^n}$.*

The proof of this theorem proceeds in three steps. The first step is to evaluate the polynomial $p^{(i)}$ (which at this point is unknown) on several random points. The second step is to do polynomial interpolation[2] on these points to recover $p^{(i)}$. Once we know $p^{(i)}$, we compute i as the discrete log of $p^{(i)}$ in the group (not group action!) of polynomials generated by p (which is easy since this group has a smooth order).

The main non-obvious part is evaluating $p^{(i)}$ at several random points. We choose several random $j \in \mathbb{Z}_{2^n}$, and compute $c_j = p^{(j)}(x)$, $d_j = p^{(j)}(y)$. Then each c_j will be uniformly random (since p forms a full-length cycle), and

$$d_j = p^{(j)}(p^{(i)}(x)) = p^{(i+j)}(x) = p^{(i)}(p^{(j)}(x)) = p^{(i)}(c_j).$$

4 Potential Next Steps

While we were able to "break" the polynomial $p^{(i)}$ by both short-cutting and solving discrete logs, we believe we have uncovered an interesting structure with the ring $\mathcal{Q}_n$. Some concrete speculative directions to explore include:

- We can characterize the group of units $\mathcal{Q}_n^*$ in $\mathcal{Q}_n$ as those where the linear coefficient is odd. $\mathcal{Q}_n^*$ is non-abelian. Can we securely plug this group into proposals for non-abelian cryptosystems?
- Cryptography has a long history of turning cryptographic attacks on their heads in order to build interesting new cryptosystems. For example, pairings were first used to break cryptosystems [MVO91], but were then turned around into novel protocols [Jou04, BF01]. Can the ability of computing discrete logarithms in $\mathcal{Q}_n$ be used to design interesting novel protocols?

[2] This is slightly non-trivial since we are not working over a field, but is nevertheless possible [MPT16].

- Fix a polynomial $p \in \mathcal{Q}_n^*$, and consider the conjugation by p: $C_p(q) = p \circ q \circ p^{-1}$. We can write this action on q as a set of polynomial equations on the coefficients of q. Note that given p, it is trivial to invert C_p; but perhaps just given the polynomial equations for C_p, inversion is difficult? If so, this could give a trapdoor permutation on $\mathcal{Q}_n$. Interestingly, this permutation is fully-homomorphic.

References

[AKK$^+$19] Hamza Abusalah, Chethan Kamath, Karen Klein, Krzysztof Pietrzak, and Michael Walter. Reversible proofs of sequential work. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 277–291. Springer, Heidelberg, May 2019.

[AS15] Joël Alwen and Vladimir Serbinenko. High parallel complexity graphs and memory-hard functions. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 595–603. ACM Press, June 2015.

[BBBF18] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 757–788. Springer, Heidelberg, August 2018.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th ACM STOC*, pages 11–19. ACM Press, May 1988.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. http://eprint.iacr.org/2006/291.

[CP18] Bram Cohen and Krzysztof Pietrzak. Simple proofs of sequential work. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 451–467. Springer, Heidelberg, April / May 2018.

[DLM19] Nico Döttling, Russell W. F. Lai, and Giulio Malavolta. Incremental proofs of sequential work. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 292–323. Springer, Heidelberg, May 2019.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[Jou04] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, September 2004.

[MMV13] Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Publicly verifiable proofs of sequential work. In Robert D. Kleinberg, editor, *ITCS 2013*, pages 373–388. ACM, January 2013.

[MPT16] G. L. Mullen, D. Panario, and D. Thomson. Fast and simple modular interpolation using factorial representation. *The American Mathematical Monthly*, 123(5):471–480, 2016.

[MVO91] Alfred Menezes, Scott A. Vanstone, and Tatsuaki Okamoto. Reducing elliptic curve logarithms to logarithms in a finite field. In *23rd ACM STOC*, pages 80–89. ACM Press, May 1991.

[Per09] Colin Percival. Stronger key derivation via sequential memory-hard functions, 2009.

[Pie19] Krzysztof Pietrzak. Simple verifiable delay functions. In Avrim Blum, editor, *ITCS 2019*, volume 124, pages 60:1–60:15. LIPIcs, January 2019.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. http://eprint.iacr.org/2006/145.

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RSW96] Ronald L Rivest, Adi Shamir, and David A Wagner. Time-lock puzzles and timed-release crypto. 1996.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

[Wes19] Benjamin Wesolowski. Efficient verifiable delay functions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 379–407. Springer, Heidelberg, May 2019.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.