IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, 34 pages.

ALFOMs and the Moirai: Quantifying the Performance/Security Tradeoff for ZK-friendly Hash Functions

Aurélien Boeuf[1] and Léo Perrin[2]

[1] Simula UiB, Bergen, Norway
[2] Inria, Paris, France

Abstract

Zero-Knowledge (ZK) protocols rely internally on hash functions for their security arguments. However, the hash functions that are the most efficient in this context differ substantially from e.g. SHA-3: their round function R must enable an efficient arithmetization of its verification. In practice, it means that verifying if $y = R(x)$ involves as few finite field multiplications as possible. In turn, this design requirement implies a greater vulnerability to algebraic attacks. In fact, their recent improvements have proven devastating, which implies the need to completely rethink the methods used to ensure security against them.

In this paper, we show that it is possible to build a simple yet efficient security argument based on a precise estimate of the so-called "ideal degree" of a system of equations. Furthermore, we show that the increase of this quantity across rounds is tightly connected to the cost of the hash function in two different arithmetizations, namely AIR and R1CS. We precisely quantify this relation by introducing ALgebraic Figures Of Merit (ALFOMs) that capture how efficient a specific primitive (and in fact its round function) is at increasing the algebraic security per unit of cost.

This new insight allows us to better understand sometimes puzzling performance differences between state-of-the-art hash functions in the R1CS and AIR cases, and to provide a fair and simple comparison of their round functions in this context. Furthermore, we present a new group of round functions we call the Moirai (a case study rather than a full new design) which allow us to explore what a round function providing optimal performance/security tradeoff could look like.

Keywords: Arithmetization-Oriented, Zero-Knowledge, R1CS, AIR, Gröbner bases

1 Introduction

Recently, there has been a surge of modern advanced protocols based on Zero-Knowledge (ZK), Multi-Party Computation (MPC) and Fully-Homomorphic Encryption (FHE) techniques. Such protocols often require basic symmetric primitives, such as block ciphers and hash functions. However, standard primitives such as AES and SHA-3 were not designed with these applications in mind, and they do not integrate well within the frameworks of these advanced protocols: they are defined over small binary fields and, while efficient to compute, their circuits have a complex arithmetic representation. By contrast, many cryptographers have recently proposed so-called Arithmetization-Oriented Primitives (AOP), that are defined over specific rings and odd-characteristic finite fields, or have simple circuit representations within specific frameworks. Indeed, a key step in the evaluation of

Emails: aurelien@simula.no (Aurélien Boeuf), leo.perrin@inria.fr (Léo Perrin)

This work is licensed under a "CC BY 4.0" license.

these primitives within the various protocols is their "arithmetization", a process whereby an arithmetic circuit is derived from the specification of the primitive. Each protocol will allow different operations within said circuits, as well as imposing a different cost for different gates.

In this paper, we narrow our focus specifically to ZK-friendly families of hash functions. This subset of AOPs has received a lot of attention in recent years, both in terms of design and in terms of cryptanalysis. Prominent examples include Rescue [AAB+20], Rescue-Prime [SAD20], RPO [AKM+22], Anemoi [BBC+23], Griffin [GHR+23] and Poseidon [GKR+21].[3] They are all designed by combining affine operations with "simple" nonlinear operations, such as power maps.

Design Requirements. Designing a ZK-friendly hash function has meant, in practice, designing a permutation which would then be used to construct a sponge [BDPVA11], or sometimes a compression function. In turn, said permutation consists in multiple iterations of a simple round function obtained by composing an affine layer with a nonlinear layer. While, at this point, nothing has come out of the ordinary of hash function design, the following additional requirements are unusual:

1. the arithmetization will be done over a specific field imposed by the protocol, typically the scalar field of some pairing-friendly elliptic curve, or the modular ring corresponding to the so-called *Goldilocks prime* $p = 2^{64} - 2^{32} + 1$ or *BabyBear prime* $p = 2^{31} - 2^{27} + 1$, meaning that the round function should be defined over a vector space of the same field,

2. the round function R must be such that it is "easy" to verify that $R(x) = y$ given both x and y.

What is "easy" depends on the specifics of the protocol and its underlying arithmetization, but a rule of thumb is that it should involve only low-degree operations over the field. However, this approximation has its limits, as we will show.

R1CS and AIR. The three main families of arithmetization for ZK protocols are the variants of Plonk [GWC19], R1CS [GGPR13] and AIR [BBHR18]. In this paper, we will leave aside Plonk as it allows for more complicated operations and finer tuning, which makes it hard to establish general statements. On the other hand, the efficiency metrics for both R1CS and AIR (whose definitions are given in Section 2.2) are rigid enough that we will be able to make strong statements about the suitability (and, in fact, the optimality) of some hash functions and their round functions for these arithmetizations. They are also somewhat similar in that the designers' goal for both is indeed a priori to limit the number of multiplications. While we leave a description of the details of these metrics for later (Section 3), we can already make some perhaps puzzling observations about the state of the art. Consider, for instance, Table 1, which is essentially lifted from [BBC+23]. It describes the cost of various ZK-friendly hash functions for different arithmetizations, and for different field size properties.

As we can see, there are some common tendencies. If the field size prevents $x \mapsto x^3$ from being bijective, in which case we are in the bottom rows, then the cost is higher, sometimes much higher. Similarly, Poseidon is never competitive. On the other hand, the Griffin/Anemoi pair on one side and Rescue-Prime on the other have opposite properties regarding AIR and R1CS: the former two are by far the best for R1CS, but they completely

[3] We purposefully leave aside hash functions based on the "Split-and-lookup" or "Kintsugi" approach, like Monolith [GKf+24]. These hash functions are based on some variants of the Plonk ZK protocol, which differ significantly from both AIR and R1CS on which we will later focus our attention.

Table 1: Performance of various ZK-friendly hash operating on $\mathbb{F}_p^4$ in different proof systems (taken from Table 2 of [BBC+23]). Smaller numbers are better. See Definitions 1 and 2 for the cost.

Condition	Proof system	Rescue-Prime	Poseidon	Griffin	Anemoi
$p-1 \not\equiv 0 \mod 3$	R1CS	224	232	112	96
	AIR	168	384	168	168
$p-1 \not\equiv 0 \mod 5$	R1CS	264	264	110	120
	AIR	220	440	220	280

lose their edge for AIR.[4] In fact, both have been the target of successful full-round attacks, meaning secure instances (using more rounds) would lose out to Rescue-Prime. In summary, this cursory study shows several things.

1. R1CS and AIR are similar enough that general tendencies across both exist.
2. On the other hand, detailed performance metrics can still vary significantly.
3. Comparing AOPs is made harder by the very quick advances in their cryptanalysis which has already made obsolete several instances that should have their round number increased.

Our goal in this paper is to solve these problems by providing a metric for each arithmetization which combines information about their efficiency *and their security*. However, in order to discuss the latter, we first recall how the cryptanalysis of AOPs works.

Polynomial System Solving. From the inception of this field of study, it has been clear that an important threat to these primitives was *Polynomial System SOlving* (PoSSo). Such techniques are used to find the solution(s) of a multivariate system of equations. As mentioned above, the round functions of AOPs must be such that verifying if $y = R(x)$ involves only the evaluation of low-degree polynomials. As a consequence, for any attack, there should exist a system of low-degree multivariate equations whose solution is what the cryptanalyst wants, such as a preimage.

As a consequence, the first iterations of Rescue were already designed in a way intended to ensure that PoSSo algorithms would have a prohibitive complexity when used to attack them. Since then, all AOP designers have provided arguments to the same effect. In practice, PoSSo is done using the following steps:

Equation writing. Define and compute a system of multivariate polynomial equations.

Obtain a univariate polynomial. If the equations are multivariate, derive a univariate equation from them. The methods for this always involve either resultants or Gröbner bases, and are summarized as follows:

1. Compute a Gröbner basis of the multivariate system for some monomial ordering using a standard algorithm such as F4 or F5.
2. Deduce a new Gröbner basis for the lexicographical order, or use the eigenvalue method [BND22] to find a univariate polynomial.

Final resolution. Find the roots of said univariate polynomial, and substitute them with the corresponding variable to find the values of the remaining variables (possibly by solving a strictly simpler multivariate system).

[4] In that claim, we account for the fact that Griffin and Anemoi are improperly scaled with respect to recent cryptanalysis. We go into more detail about this issue in later sections.

Devastating Attacks. The security of most AOPs against such attacks has been argued by estimating the cost of the Gröbner basis computation, and ensuring that the number of rounds was high enough to make it unfeasible. Unfortunately, several recent breakthroughs have shown this approach to be insufficient. Resultant-based attacks [YZY+24, BBB+25] and Gröbner basis attacks, such as the Freelunch attacks [BBL+24] or dedicated approaches to compute Gröbner bases efficiently [ACG+19, CR25] turned out to break full-round instances of Griffin, Anemoi, Arion and Rescue, far past the analysis of their designers. We identify two main reasons for this:

1. The designers of these primitives all relied on the hardness of computing Gröbner bases of the polynomial models to compute their round numbers.

2. Computing a Gröbner basis of these models is in some cases cheap (or even free in the case of the Freelunch attacks) or not required altogether in the case of resultant-based attacks.

A better approach is thus needed to ensure the security against such attacks.

On the Importance of the "Ideal Degree". All of these attacks still involve a classical quantity in commutative algebra that is a fundamental attribute of the polynomial model: the degree of the ideal spanned by the multivariate polynomials, denoted D_I. This quantity corresponds to the dimension, as a vector space, of the multivariate ring modulo the ideal [CLO25]. The complexity of the attacks described above is at least linear in D_I, and at most cubic in it. Furthermore, and crucially, D_I does not depend on the intricacies of how the system is obtained. It is in fact an inherent property of the attack being modeled. As such, using a different method to derive the equations will not modify it.

The only techniques known to reduce D_I are the so-called "round skips", which, as their name implies, leverage particular structures at the input or output to avoid needing to define a few variables or equations within the first few rounds, effectively skipping (parts of) them.

We therefore argue that convincing security arguments against PoSSo-based attacks can be made by lower-bounding the ideal degree corresponding to various attacks.

Our contributions. The key insight in this work is a surprising relationship between D_I and the performance of an AOP in either R1CS or AIR. Indeed, when comparing the growth of D_I with the number of rounds in state-of-the-art PoSSo cryptanalysis, and the increase in cost of the corresponding primitives within either R1CS or AIR with the number of rounds, clear, simple relations emerge. We call these *ALgebraic Figures Of Merit* (ALFOMs).

In this paper, we formally define ALFOMs for AIR and R1CS. They are meaningful and simple metrics, which allow us to evaluate the design of existing ZK-friendly hash functions in a unified, coherent way by capturing the performance/security trade-off of their round function. This approach solves most the problems we identified above.

- The security level of AOPs is easily computed using this method, meaning that it is easy to update the number of rounds of broken primitives to provide a given security level. A comparison at an equivalent security level with respect to this attack then becomes possible.

- The similarities and differences between R1CS and AIR are captured by their differing ALFOMs. In fact, we show that the ALFOM of Rescue-Prime for R1CS is much worse than that of Anemoi or Poseidon, while the exact opposite is true for AIR: the discrepancies in Table 1 are directly explained using this tool.

Beyond a better understanding of the state-of-the-art, we also show that ALFOMs can serve as guiding principles when building round functions. In fact, we can even argue that a round function is *optimal*, in the sense that it allows the best possible increase of the ideal degree for a given AIR or R1CS cost. We showcase this possibility by introducing a new group of ZK-friendly hash functions we call the Moirai: Clotho, Lachesis and Atropos. These constructions allow us to explore the limits of maximizing ALFOMs across both R1CS and AIR. To be clear, they are intended as a case study rather than as a complete new design proposal.

Outline. In Section 2, we recall the definition of some ZK-friendly families of permutations used to design hash function, their cost models and state-of-the-art cryptanalysis methods used against them. In Section 3, we introduce the ALFOMs metrics, justify their definition and relevance, and we give some optimal bounds on them. In Section 4, we formally evaluate the ALFOMs of Rescue, Poseidon and Anemoi, and comment their designs in the light of their ALFOMs. Lastly, in Section 5, we introduce the Moirai, constructions that were conceived to maximize ALFOMs (and reach optimal values) upon various constraints.

2 ZK-friendly hash functions and their analysis

2.1 Some ZK-friendly hash functions

In this section, we describe the families of permutations Anemoi [BBC+23], Rescue-Prime [SAD20] and Poseidon [GKR+21], which we will study in more detail in later sections. All of these are used to instantiate hash functions using the sponge construction.

2.1.1 Poseidon

Poseidon is a family of ZK-friendly permutations based on the simple $x \mapsto x^\alpha$ permutations (for α coprime to $p-1$). In this paper, we will mainly consider Poseidon [GKR+21], rather than Poseidon2 [GKS23], the latest iteration, because more detailed cryptanalysis exists for it, though we expect Poseidon2 to have a very similar analysis. Poseidon is defined over $\mathbb{F}_p^b$ for $b \geq 2$. One permutation consists of R_F "full" rounds and R_P "partial" rounds: a full round's nonlinear layer applies the $x \mapsto x^\alpha$ permutation on every branch, while a partial round only applies it on one branch. A figure of the Poseidon permutation is given in Figure 3. Mathematically, it has the following shape:

$$\underbrace{\mathcal{M} \circ \mathcal{S}_\mathcal{F} \circ ARC_{R_F+R_P} \circ \ldots}_{R_F/2 \text{ rounds}} \circ \underbrace{\cdots \circ \mathcal{M} \circ \mathcal{S}_\mathcal{P} \circ ARC_j \circ \ldots}_{R_P \text{ rounds}} \circ \underbrace{\cdots \circ \mathcal{M} \circ \mathcal{S}_\mathcal{F} \circ ARC_1}_{R_F/2 \text{ rounds}},$$

where $\mathcal{S}_F$ is the application of $x \mapsto x^\alpha$ on every branch, $\mathcal{S}_P$ is its application on only one branch, $\mathcal{M}$ denotes multiplication by a given matrix and ARC_j is the round constant addition of the j-th round.

2.1.2 Rescue

For this paper we focus on Rescue-Prime [SAD20], as it is the most studied member of the Rescue family. Rescue-Prime features the power maps $x \mapsto x^\alpha$ and $x \mapsto x^{1/\alpha}$ as its Sboxes, where $\alpha = \min\{\alpha' : \gcd(\alpha', p-1) = 1\}$. The round function of Rescue-Prime is given in Figure 4. The full permutation is described as

$$\mathcal{A}_{r,1} \circ \mathcal{S}^{-1} \circ \mathcal{A}_{r,0} \circ \mathcal{S} \circ \cdots \circ \mathcal{A}_{1,1} \circ \mathcal{S}^{-1} \circ \mathcal{A}_{1,0} \circ \mathcal{S}$$

where $\mathcal{S}$ denotes the application of $x \mapsto x^\alpha$ on every branch, and $\mathcal{A}_{j,0}$ and $\mathcal{A}_{j,1}$ are the two affine layers of the j-th round, $1 \leq j \leq r$.

2.1.3 Anemoi

Anemoi [BBC+23] is a family of permutations that has two main variants: one in even characteristic and one in odd characteristic. We will mainly study the odd characteristic case, as it is the only one for which the original authors gave benchmarks. It is also the most frequently investigated case. The number of branches b is always taken as an even integer (due to the Feistel-like structure of the nonlinear layer). Anemoi can be decomposed as follows:

$$\mathcal{A}_r \circ \mathcal{S} \circ \cdots \circ \mathcal{A}_1 \circ \mathcal{S} \circ \mathcal{A}_0,$$

where the $\mathcal{A}_j$ are affine layers (their full description is a bit complicated, we refer to the original paper) and $\mathcal{S}$ denotes the successive application of the following maps (a so-called "Flystel") on each pair of branches:

$$((x,y) \mapsto (x + gy^2, y)) \circ ((x,y) \mapsto (x, y - x^{1/\alpha})) \circ ((x,y) \mapsto (x - gy^2 - g^{-1}))$$

where g is a given element of $\mathbb{F}_p$. Figure 5 represents the first few layers.

2.2 On ZK Constraint Systems and their Cost Functions

In this section, we give an overview of the Zero-Knowledge constraint systems R1CS and AIR, and we give a reminder on how designers have been able to evaluate how efficient primitives are relative to each of these constraint systems through cost functions.

2.2.1 R1CS

In the context of Rank One Constraint Systems (R1CS) [PHGR13], the permutation is modeled as a multivariate polynomial system where each equation has one of these forms:

$$\begin{cases} z = x + y & \text{where } z \text{ is a variable and } x, y \text{ are variables or constants,} \\ z = x \cdot c & \text{where } x, z \text{ are variables and } c \text{ is a constant, or} \\ z = x \times y & \text{where } x, y, z \text{ are variables.} \end{cases}$$

Such a polynomial system is then translated as a constraint system in the context of a Zero-Knowledge protocol. Out of these three types of equations, the variable-variable multiplications ends up being, by far, the most costly ones in the context of the protocol. Therefore, a permutation that is efficient within the context of R1CS is a permutation that can be modeled with a "small" number of multiplications.

Note that a primitive of low degree, such as MiMC [AGR+16] or Poseidon [GKR+21], will automatically yield low-degree models. However, it is perfectly possible to mix operations which either have low degree or have an inverse of low degree. This strategy is prominently used in [SAD20], where the Sboxes $x \mapsto x^\alpha$ and its inverse (for a low value of α, such as $3, 5$ or 7) are used.

Definition 1 (R1CS Cost)**.** Let P be a permutation. The R1CS Cost of P is denoted $C_{\text{R1CS}}(P)$ and corresponds to the number of multiplications in a given R1CS model of P.

Example 1 (Rescue-Prime)**.** In Rescue-Prime [SAD20], the number of Sboxes is $2 \cdot r \cdot b$, where b is the number of branches and r is the number of rounds. Consider a Rescue-Prime permutation P for any fixed p, r and b and define α as the smallest integer such that $x \mapsto x^\alpha$ is a permutation. Each Sbox is either $x \mapsto x^\alpha$ or its functional inverse $x \mapsto x^{1/\alpha}$, where $1/\alpha$ is the multiplicative inverse of α modulo $p - 1$. The authors of Rescue-Prime compute the R1CS Cost as follows: each Sbox can be evaluated with a certain number of multiplications which depends on α (see Section B), which we write as $\ell(\alpha)$. Then, the R1CS Cost is

$$C_{\text{R1CS}}(P) = 2 \cdot \ell(\alpha) \cdot r \cdot b.$$

2.2.2 AIR

The Algebraic Intermediate Representation (AIR) is a kind of arithmetization that was conceived for the STARK protocol, along with primitives like Rescue-Prime. It consists of the following elements:

1. A set of $t \cdot w$ variables $\{x_{i,j}\}$ where t is the number of steps of the evaluation, and w is the "width", the number of registers that we specify at each step (typically, the values taken by the branches of a block cipher after i rounds).
2. A secret matrix of size $t \cdot k$, called the "execution trace", where the coefficient i, j is the value taken by $x_{i,j}$ (the j-th register at step i).
3. A public set of polynomial constraints on variables $\{x_{i,j}\}$.

Definition 2 (AIR Cost)**.** Let P be a permutation, and consider an AIR model of P with $t \cdot w$ variables such that $d_{\max}$ is the maximum total degree of the polynomial constraints. Following [AAB+20], the AIR Cost of P is $C_{\text{AIR}}(P) = t \cdot w \cdot d_{\max}$.

Example 2 (Rescue-Prime AIR)**.** For Rescue-Prime, $x_{i,j}$ is essentially the value of the j-th variable after i rounds. The specifics may vary, but for b branches and r rounds, we typically get a set of $r \cdot b$ polynomial constraints of total degree α, where $\alpha = \min\{\alpha' : \gcd(p-1, \alpha') = 1\}$. Then, the AIR Cost is simply $r \cdot b \cdot \alpha$.

2.3 Attacking a Hash Function using PoSSo

In the absence of a "secret key", the security of public permutation families used in sponge constructions [SAD20, AKM+22, GHR+23, BBC+23, MA23] is often formulated in terms of the hardness of the CICO problem.

Definition 3 (CICO problem)**.** Let $\mathbb{F}$ be a field and n, k be positive integers. Given a permutation $\mathbf{P} : \mathbb{F}^n \to \mathbb{F}^n$, we define the CICO-$k$ problem as follows: find $\mathbf{x} \in \mathbb{F}^n$ such that both $\mathbf{x} \in \mathbb{F}^{n-k} \times \{0\}^k$, and $\mathbf{P}(\mathbf{x}) \in \mathbb{F}^{n-k} \times \{0\}^k$.

A brute-force strategy to solve CICO-k for $\mathbf{P}$ consists in evaluating $\mathbf{P}$ on random inputs $\mathbf{x} \in \mathbb{F}^{n-k} \times \{0\}^k$ until we find an output $\mathbf{P}(\mathbf{x}) \in \mathbb{F}^{n-k} \times \{0\}^k$. If $\mathbb{F}$ has size q and if $\mathbf{P}$ is a random permutation, then we can expect this approach to require q^k evaluations of $\mathbf{P}$ on average. On the other hand, a common approach to find solutions to this problem in a more efficient way consists in reducing CICO problems to PoSSo problems. Such techniques fall under the (very broad) umbrella of *algebraic cryptanalysis*. In the following sections, we describe the main CICO-to-PoSSo reductions currently used in cryptanalysis.

2.3.1 Univariate Root-Finding

Univariate root-finding algorithms can be used to solve the CICO-1 problem. It performs better than brute-force under the condition that the permutation considered has a sufficiently low degree. The expected number of solutions is easily established using the results in [FO90], and summarized in Proposition 1.

Lemma 1 (CICO-1 to Univ)**.** *Let $\mathbf{P} : \mathbb{F}^n \to \mathbb{F}^n$ be a permutation. Consider $\mathbf{P}$ restrained to any affine space A of dimension 1 parametrized by a single variable x in $\mathbb{F}_q$, that verifies the following equation:*

$$A = \{x\mathbf{v} + \mathbf{a} : x \in \mathbb{F}_q\} \subset \mathbb{F}_q^{n-1} \times \{0\}.$$

Let $\mathbf{P}_{\mathbf{v},\mathbf{a}} : \mathbb{F}_q \to \mathbb{F}_q$ be the function that maps $x\mathbf{v} + \mathbf{a}$ to the n-th coordinate of its image by $\mathbf{P}$. Then, a root of this function is a solution to the CICO-1 problem.

Proposition 1 (Probability of finding a root)**.** *The probability that a random function $f : \mathbb{F}_q \to \mathbb{F}_q$ has a root is $Pr_f[\exists x \in \mathbb{F}_q : f(x) = 0] = 1 - (1 - 1/q)^q$. which converges to $1 - 1/e \approx 0.63$ as $q \to +\infty$. More generally, the number of roots of a random function follows a Poisson law of parameter 1.*

In order to find the roots of the given univariate polynomials, rather than directly applying the general Berlekamp [Ber71] or Cantor-Zassenhaus [CZ81] algorithms, one can first compute their greatest common divisor with the field polynomial $x^q - x$ using a fast half-GCD [Moe73] algorithm. This will yield a polynomial whose degree follows a Poisson distribution of parameter 1 (see Proposition 1). That polynomial has average degree 1, and can be factorized in a negligible time (compared to the time it takes to compute the gcd). This method is used in state-of-the-art algebraic cryptanalysis [BBLP22, BBL+24]. Thus, the complexity of computing the roots of a random polynomial of degree d in $\mathbf{F}_q$ is

$$\mathcal{O}\left(d\log(d)\left(\log(d)+\log(q)\right)\log\left(\log(d)\right)\right) \ . \tag{1}$$

Efficient univariate root-finding is frequently used in various PoSSo algorithms, even in the multivariate case. The problem reduction from Lemma 1 is particularly relevant for the cryptanalysis of arithmetization-oriented primitives whose round function has low degree, such as MiMC [AGR+16] or Poseidon [GKR+21, GKS23].

Remark 1 (CICO-k to CICO-1 reduction). Suppose that you can solve the CICO-1 problem faster than brute-force using the reduction from Lemma 1 for approximately q^{k-1} distinct affine spaces A such that each of these affine spaces is included in $\mathbf{F}_q^{n-k} \times \{0\}^k$. Then, it is possible to solve the CICO-k problem by computing a solution to the CICO-1 problem where the input is in $\mathbf{F}_q^{n-k} \times \{0\}^k$. Then, there is a probability of $1/q^{k-1}$ that the $k-1$ "leftover" coordinates are all equal to 0, which would make this CICO-1 solution a CICO-k solution.

2.3.2 Gröbner Basis Solving

For ZK-friendly hash functions, it is the *verification* that needs to correspond to low-degree polynomials, i.e. the operation taking the (x, y) as input that returns true if $y = R(x)$. Thus, designers (starting with those of Rescue [AAB+20]) have relied not only on low-degree functions, but also on their functional inverse. This trick means that the permutation quickly reaches a very high degree, rendering the above approach useless. However, it is then possible to introduce intermediate variables and new equations to obtain a multivariate system only containing low-degree equations.

Definition 4 (CICO-k to Gröbner Basis Solving)**.** Let $\mathbf{P} : \mathbb{F}^n \to \mathbb{F}^n$ be a permutation and $k < n$. Consider $\mathbf{P}$ restrained to any affine space A of dimension k parametrized by k variables $x_1, \ldots, x_k$ in $\mathbb{F}_q$, that verifies the following equation for some vectors $\mathbf{v_i}$ and $\mathbf{a}$:

$$A = \left\{ \sum_{i=1}^{k} x_i \mathbf{v_i} + \mathbf{a} : x_1, \ldots, x_k \in \mathbb{F}_q \right\} \subset \mathbb{F}_q^{n-k} \times \{0\}^k.$$

Suppose that we are able to introduce intermediate variables $y_1, \ldots, y_r$ such that there exists a polynomial system

$$\begin{cases} P_1(x_1, \ldots, x_k, y_1, \ldots, y_r) &= 0\,, \\ \vdots & \\ P_m(x_1, \ldots, x_k, y_1, \ldots, y_r) &= 0\,, \end{cases}$$

whose solutions are CICO-k solutions (this constraint dictates the definition of the polynomials). Then, we can use methods such as Gröbner basis solving to find solutions of this system in $\mathbb{F}_q$ and thus solve CICO-k.

Complexity of Gröbner Basis Solving. Consider a polynomial system like the one in Definition 4, and suppose that the ideal spanned by the P_i for $1 \leq i \leq m$ has dimension 0.[5] Then, solving that system is done in the three main steps, as described in the introduction: the first two are intended to find a univariate polynomial, while the last is a univariate root-finding step. The first step consists in computing a Gröbner Basis of the system for any monomial order using an algorithm such as F4 [Fau99] or F5 [Fau02], and the second one consists in changing this monomial order to *lex* using an algorithm such as FGLM [FGLM93]. These first two steps are the bottleneck in practice as the last step can be done in quasi-linear time (see Section 2.3.1).

Some authors [BBC+23, SAD20] explicitly try to evaluate the time complexity of finding a Gröbner basis using the F4 or F5 algorithms, and use this to determine the number of rounds needed for a given security level. However, it has been recently shown [BBL+24, CR25] that, in many cases, it can be much easier (or even "free") to find a Gröbner Basis in a specific monomial ordering. In this case, the second step becomes the bottleneck. Its complexity is formulated as a function of the degree of the ideal, D_I, a very important quantity which we will discuss in Section 3.1 in more detail.

Proposition 2 (FGLM complexity)**.** *Let I be an ideal of dimension 0 of $\mathbb{F}[x_1, \ldots, x_n]$ and G be a Gröbner Basis for I in any monomial ordering $\prec$. Let D_I be its degree. Then the FGLM algorithm [FGLM93] computes a Gröbner Basis in lexicographical ordering with a time complexity of $\mathcal{O}(nD_I^3)$.*

3 Introducing ALFOMs

3.1 On the Importance of the Ideal Degree

In Section 2.3, we described a general approach to solve polynomial systems of equations. The FGLM algorithm is generic but, in some cases, there exist better ways to compute a univariate polynomial from a PoSSo instance, such as Sparse-FGLM variants [FM17], dedicated optimized algorithms [BBL+24], or even resultant-based methods [BBB+25, YZY+24]. All these methods can yield a univariate polynomial without the need of a Gröbner Basis. However, all of these methods depend at least quasi-linearly in the degree of the ideal, D_I. Note that under generic assumptions, D_I is also the smallest degree of all univariate polynomials in the ideal. As a consequence, we make the following claim, which will serve as the basis for the remainder of this paper.

Claim. *The ideal degree is the most important quantity to study when evaluating the hardness of PoSSo-based cryptanalysis in Arithmetization-Oriented primitives, and a lower bound on its value is a sufficient security argument against PoSSo-based attacks.*

It is worth mentioning that the authors of XHash provided some security arguments based on conjectured lower bounds on D_I [ABK+23, Conjecture 1].

3.2 Quantifying the Performance/Security Tradeoff

This claim motivates our definition of a metric that defines the security-to-performance ratio of AOPs for a given arithmetization using:

- D_I as a representative of the "security" of a public permutation against PoSSo cryptanalysis, and
- a relevant cost function for the given arithmetization as a representative of the "performance".

[5] This condition is needed by the algorithms described further. Its specific definition is too involved for this discussion, but is satisfied in our case as we expect the system to have 1 or a couple solutions.

Moreover, we observe that in a lot of relevant works, D_I grows exponentially with the number of rounds, whereas the number of constraints in the underlying ZK-protocol grows linearly with the number of rounds. Thus, we can define a metric for the overall efficiency of a public ZK-friendly permutation that is invariant in the number of rounds that we call Arithmetic Efficiency.

Definition 5 (ALFOMs)**.** Let $P(r)$ be a family of permutations defined for any number r of rounds. Fix an integer k and suppose that, for any $r \geq 1$, we can derive a PoSSo instance of the CICO-k problem spanning an ideal whose degree is denoted by $D_I(r, k)$. Let the cost function in the underlying ZK protocol be $C(r)$. Then, we define the *Algebraic Figures Of Merit* (ALFOMs) of this family as follows:

$$\eta(P(r), k) = \frac{\log_2(D_I(r, k))}{C(r)}.$$

Furthermore, assume that there exist d_k and c such that for all $r \geq 1$, $D_I(r, k) = d_k^r$ and $C(r) = rc$. Then, $\eta(P(r), k)$ is constant in r, and we simply write $\eta(P, k) = \log_2(d_k)/c$.

If the ALFOMs do not depend on the number of rounds, then it is in fact a property of the round function of the primitive. This allows us to compare the actual efficiency of some ZK-friendly families of permutations simply by looking at their round functions. While this approach is broad and loses some nuance, such as round skips and how the number of rounds is defined, we will show in the rest of this section that it proves relevant in practice, and we will examine how it can guide design. In particular, in Section 3.3, we will define optimality bounds in the context of R1CS and AIR-related ZK systems of constraints, with the associated "optimal" designs according to this metric.

Choice of Polynomial System. In the previous definition, D_I is dependent on the polynomial systems chosen by the cryptanalyst (and in particular, depends on k). In the simplest cases, D_I will have the form d_k^r. However, the cryptanalyst has some leverage to decrease D_I: when solving a CICO-k problem, they can choose a subset of inputs which does not activate some nonlinear layers. This is what we call "round skips" in the context of PoSSo-based cryptanalysis. When modeling the action of P on such inputs, some equations can therefore be removed, or have lower degree, or fewer monomials, which can decrease D_I and the complexity of the overall attack. It is generally an open question to determine the models that achieve the smallest D_I.

Relation between ALFOMs and full cost. In the case of AOPs where the best cryptanalysis (aside from brute force) tends to come from PoSSo-based attacks, there is a direct relation between ALFOMs and the ZK-cost of the whole hash function. Indeed, the number of rounds is chosen so that the best known algorithms to solve CICO-k for a certain k (typically half the capacity of the sponge) are greater than a given security bound 2^λ. Fix a ZK system and consider the simple case where we can write $\eta(P, k) = \log_2(d_k)/c$. When the number of rounds is chosen[6] so that $\log_2(D_I(r, k)) \approx \lambda$, we can write the cost function C as a function of λ and we have that $C(\lambda) \approx \lambda/\eta(P, k)$.

More formally, assume that the number of rounds is computed as $r = \lceil \lambda / \log_2(d_k) \rceil + \gamma$ where γ is the number of bonus marginal rounds. We have that as λ increases,

$$C(\lambda) = \frac{\lambda}{\eta(P, k)} + o(1) \ .$$

[6]Which is true for Poseidon instances where $k = 1$, but an "approximation error" appears since a few rounds are always added to account for round skips and to give some security margin.

3.3 Optimality bounds

ALFOMs cannot take arbitrary values: as we will show, there are upper bounds, meaning that there is such a thing as an *optimal* ZK-friendly hash function—at least in theory. In what follows, we derive upper bounds for ALFOMs for both R1CS and AIR. There are two main cases to consider. The first is the "Poseidon-like" case, where all of the components of P (or P^{-1}) have low degree. In this case, to model a CICO-k instance, one typically only needs to introduce k variables at the input (resp. output). The second is the "Rescue-like" case, where there is an alternating use of low-degree and low-degree-inverse components, which forces the presence of many intermediate variables in the PoSSo instance.

3.3.1 R1CS ALFOMs Bounds

Recall from Definition 1 that the R1CS-cost is the number of multiplications in the underlying model. So, for R1CS, the ALFOM is the ratio between $\log_2(D_I)$ and the aforementioned quantity. With m multiplications, the greatest power of x that can be achieved[7] is x^{2^m}. In fact, we deduce from the classical Bézout bound that, given an R1CS system with m multiplications (and thus a polynomial system of equations), D_I can be no more than 2^m, no matter which CICO-k instance is being studied. Thus, we get the following proposition.

Proposition 3 (R1CS bound)**.** *If a permutation P can be arithmetized in R1CS then, for any k, $\eta_{R1CS}(P,k) \leq 1$.*

Proof. Any dimension-0 model whose variables appear in the R1CS has at most as many solutions as the "naive" models where every atomic operation has an equation, and whose equations are therefore a subset of the equations that appear in the R1CS model. Furthermore, let m be the number of multiplications appearing in this model. Polynomial systems used to attack this primitive would have at most m equations of degree 2, while all the others have degree 1. By the Bézout bound, we have that $D_I \leq 2^m$. Thus, we deduce that $\log_2(D_I) \leq m$, and that $m = C_{R1CS}(P)$. □

Optimal R1CS constructions. Given the existence of an upper bound, it is natural to ask whether it is tight. Are there optimal constructions in the sense that, to solve any CICO-k on any number of rounds, $\eta_{R1CS}(P(r,k)) = 1$? This question is not easy to answer due to the existence of the "round skips" mentioned in Remark 3.2. However, under the hypothesis that the decrease of D_I due to round skips is upper-bounded, it is possible to define constructions whose ALFOM converges to 1 as the number of rounds increases using quadratic nonlinear functions (see Section 5). In fact, the importance of quadratic functions in this context is well-known by designers, and they have been extensively studied [GOPS22, GGOP23].

3.3.2 AIR ALFOMs Bounds

Much like with R1CS, the existence of an efficient AIR modeling directly yields an upper bound on D_I. We recall that, for a permutation P, $C_{AIR}(P) = w \cdot t \cdot d_{\max}$. With those notations, an AIR model consists of $w \cdot t$ equations of total degree at most $d_{\max}$ in variables that represent values taken by the elements of the state at different steps of the computation. From those, one can derive a dimension-0 ideal to model any CICO problem, where the Bézout bound gives $D_I \leq d_{\max}^{w \cdot t}$. We deduce the following proposition.

[7] A more thorough discussion on this topic involves so-called *addition chains* and is provided in Appendix B.

Proposition 4 (AIR bound)**.** *If a permutation P can be modeled by an AIR containing equations of degree at most $d_{\max}$, then, for any value of k,*

$$\eta(P,k) \leq \frac{\log_2(d_{\max})}{d_{\max}} .$$

The maximum is reached for $d_{\max} = 3$, in which case $\eta(P,k) \leq \log_2(3)/3 \approx 0.53$.

4 Case Studies: the ALFOMs of some primitives

In this section, we give explicit values for the ALFOMs coefficients for the families of primitives that we presented in Section 2.1: Rescue, Poseidon, and Anemoi for both R1CS and AIR. The D_I values are taken from the best existing analysis and round skip methods. Most importantly, we do not just compute ALFOMs for fixed parameters: we give simple expressions for their asymptotic behavior which show a lot more clearly why some primitives are better suited for certain ZK systems than others. For example, it will be very clear why Rescue is excellent in AIR (which it was designed for), but not in R1CS.

4.1 Rescue

For this section we focus on Rescue-Prime, whose definition we recalled in Section 2.1. Let r be the number of rounds and b be the number of branches. For any k, solving the CICO-k problem involves a system of $k + b \cdot r$ equations and variables. Of those equations, $b \cdot r$ of them have degree α (they correspond to the intermediate variables added after $x \mapsto x^{1/\alpha}$ and k of them are linear. Thus, the Bézout bound gives us the following, for any k:

$$D_I \leq \alpha^{b \cdot r}. \tag{2}$$

Experimental results from [BBLP22, Table 5] suggest that this bound is, in fact, tight. So, it will be the value we use, accounting for round skips. A 1-round skip is described in [BBLP22, Section 4.4] in the case where $b = 3$ and $k = 1$. A more general way to skip the first round is described in [BCP23]. Note that this is possible due to the fact that in Rescue-Prime, the first layer is the Sbox layer, which does not make the CICO problems harder at all: this mistake was changed in the more recent descendants of Rescue, RPO [AKM+22] and XHash [ABK+23]. Therefore, we take $D_I = \alpha^{b \cdot (r-1)}$.

Recall from Example 1 that $\ell(\alpha)$ is the smallest number of multiplications required to compute x^α in the R1CS model (see Appendix B for more details). The cost functions for Rescue are as follows:

$$\begin{aligned} C_{R1CS}(\alpha, b, r) &= 2 \cdot \ell(\alpha) \cdot b \cdot r, \\ C_{AIR}(\alpha, b, r) &= \alpha \cdot b \cdot r. \end{aligned}$$

Now, taking the first round skip into account, we can compute the ALFOMs:

$$\begin{aligned} \eta_{R1CS}(\alpha, b, r) &= \frac{\log_2(\alpha)}{2 \cdot \ell(\alpha)} \left(1 - \frac{1}{r}\right), \\ \eta_{AIR}(\alpha, b, r) &= \frac{\log_2(\alpha)}{\alpha} \left(1 - \frac{1}{r}\right). \end{aligned}$$

The $1/r$ in these expressions is due to the first round skip. We observe that:

- The ALFOMs do not depend on the number of branches, and the part of the expression dependent on r converges to 1.

- In the R1CS case, the $\log_2(\alpha)/\ell(\alpha)$ factor is inherent to the fact that we are not using quadratic functions, and therefore ALFOMs cannot reach the optimal value of 1 (see Appendix B). More importantly, the 1/2 factor is essentially due to the fact that, for b equations of degree α, there are $2b$ Sboxes: b of degree α and b of degree $1/\alpha$. This is an inherent inefficiency of Rescue (in the R1CS case, as Rescue was not designed specifically for it), which approximately translates to double the constraints compared to a theoretical optimal permutation of high degree.
- For R1CS, $\log_2(\alpha)/\ell(\alpha) < 1$, but some values of α can make it approach 1.
- On the other hand, in the AIR case, there is no factor 1/2. The term α instead of $\ell(\alpha)$ might be surprising at first, as $\ell(\alpha)$ is more comparable to $\log_2(\alpha)$, but recall that the cost functions for R1CS and AIR are of a different nature and shouldn't be compared directly. Note that smaller values of α give better ALFOMs, and the most efficient value of α for Rescue is 3. There is indeed a clear difference between $\alpha = 3$ and $\alpha = 5$ in [BBC+23, Table 2]. In fact, for $\alpha = 3$, without round skips, Rescue is AIR-optimal (see Proposition 4).

4.2 Poseidon

In this section, we will mainly consider Poseidon [GKR+21], whose definition we recalled in Section 2.1, rather than Poseidon2 [GKS23], the latest iteration, because more detailed cryptanalysis exists for it, though we expect Poseidon2 to have a very similar analysis. While every round has total degree α, this does not necessarily mean that, when solving a CICO-k problem, we have $D_I = \alpha^{k(R_F+R_P)}$. The authors of [BGL20] construct a system where we have

$$D_I = \alpha^{k(R_F-2)+R_P+1}.$$

The $(R_F - 2)$ term comes from the fact that, essentially, two rounds are skipped. We expect that in Poseidon2, where the authors added an additional linear layer to prevent one round skip, this term would be replaced by $(R_F - 1)$ (although a rigorous justification lies beyond the scope of this paper).

For the number of constraints, the R1CS complexity is equal to the number of Sboxes times $\ell(\alpha)$. A good AIR complexity requires some work: we do have that $w = b$ and $d_{\max} = \alpha$, but T can be reduced to $R_F + \lceil R_P/b \rceil$ compared to the naive $R_F + R_P$ (see [BBC+23, Section 7.3]). Thus, we have that

$$\begin{aligned} C_{R1CS}(\alpha, b, R_F, R_P) &= \ell(\alpha) \cdot (b \cdot R_F + R_P), \\ C_{AIR}(\alpha, b, R_F, R_P) &= \alpha \cdot b \cdot \left(R_F + \left\lceil \frac{R_P}{b} \right\rceil \right). \end{aligned}$$

Putting everything together, we get the ALFOMs:

$$\begin{aligned} \eta_{R1CS}(\alpha, b, R_F, R_P, k) &= \frac{k(R_F-2)+R_P+1}{bR_F+R_P} \cdot \frac{\log_2(\alpha)}{\ell(\alpha)}, \\ \eta_{AIR}(\alpha, b, R_F, R_P, k) &= \frac{k(R_F-2)+R_P+1}{bR_F+b\lceil R_P/b \rceil} \cdot \frac{\log_2(\alpha)}{\alpha}. \end{aligned}$$

- For both R1CS and AIR, the ALFOMs increase with k, and decrease with b. This is natural: the fewer input variables we have, the more full rounds are "overkill". For instance, to solve CICO-1, the output (univariate) polynomial has degree $\alpha^{R_F+R_P}$ (ignoring round skips): full rounds count as much as partial rounds for security, yet full rounds cost much more. At the same time, the authors do not claim that CICO-1 is secure for the security parameter λ. We argue that the most relevant value of k in this context is the smallest k such that $p^k > 2^\lambda$ (generally, k is the capacity of the sponge).

- The most interesting "asymptotic" behavior is when R_P tends to infinity. Indeed, the Poseidon (and Poseidon2) permutations are scaled so that the number of full rounds is small (constant in fact: $R_F = 8$ on all proposed concrete instances) compared to the number of partial rounds, which grows according to p, b, α and λ. And indeed, we have that:

$$\begin{aligned} \eta_{R1CS}(\alpha, b, R_F, R_P, k) &\underset{R_P \to \infty}{\sim} \frac{\log_2(\alpha)}{\ell(\alpha)}, \\ \eta_{AIR}(\alpha, b, R_F, R_P, k) &\underset{R_P \to \infty}{\sim} \frac{\log_2(\alpha)}{\alpha}. \end{aligned}$$

  This asymptotic behavior is, as far as we know, as good as it gets for primitives based on power-α Sboxes. And indeed, the partial rounds do achieve the "ideal" scenario where each Sbox adds an α factor to D_I. However, for practical instances, the full rounds do introduce a certain amount of inefficiency (see Table 3). Furthermore, each round has direct "low-degree", which may introduce some additional attack vectors compared to a primitive like Rescue: a recent cryptanalytic work [GKR25] has exploited Neptune's (worse) property, that all rounds and their compositional inverse have low degree.

4.3 Anemoi

Anemoi is the last family of permutations that we will study. Its definition was recalled in Section 2.1. From the original paper [BBC+23], the cost functions are:

$$\begin{aligned} C_{R1CS}(\alpha, b, r) &= (\ell(\alpha) + 2) \cdot \tfrac{b}{2} \cdot r\,, \\ C_{AIR}(\alpha, b, r) &= \alpha \cdot b \cdot r\,. \end{aligned}$$

In the case $b = 2$ (in which case only CICO-1 instances have solutions), we use $D_I = (\alpha+2)^r$. This value is experimentally tight and is generally an upper bound as proven in [Bri23, Chapter 11], where the authors conjecture that for the general case, r can be replaced by $r \cdot \frac{b}{2}$. Round skips are described in [BBB+25] when $b \geq 4$, where $\frac{b}{2} - 1$ equations and variables are removed for solving the CICO-1. It is possible to generalize these skips for CICO-k for any k, where we can remove $\frac{b}{2} - k$ equations (see [BP25, App. D]), and so we take, for all parameters:

$$D_I = (\alpha + 2)^{\frac{b}{2}(r-1)+k}\,.$$

Notice that the previous formula also works for $b = 2$ and $k = 1$ even though there are no round skips in that case. We get the following ALFOMs formulae:

$$\begin{aligned} \eta_{R1CS}(\alpha, b, r, k) &= \frac{\log_2(\alpha + 2)}{\ell(\alpha) + 2} \left(1 + \frac{1}{r}\left(\frac{k}{\frac{b}{2}} - 1\right)\right), \\ \eta_{AIR}(\alpha, b, r, k) &= \frac{\log_2(\alpha + 2)}{2\alpha} \left(1 + \frac{1}{r}\left(\frac{k}{\frac{b}{2}} - 1\right)\right). \end{aligned}$$

Note that the whole part in parentheses converges to 1 as r tends to infinity. While round skips shouldn't be ignored, this implies that the most important parts of these formulas are the $\log_2(\alpha)/(\ell(\alpha) + 2)$ and $\log_2(\alpha)/(2\alpha)$. There are a few observations we can make:

- In the R1CS model, Anemoi's ALFOM can converge to the optimal value of 1, as r tends to infinity and we choose $\alpha = 2^k + 1$ as k tends to infinity. However, practically, only very small values of α are chosen, and the ALFOMs of Anemoi are hindered by the $+2$ in the numerator which is inside of the $\log_2$ rather than outside. This is due

to the fact that each round of Anemoi has $2\ell(\alpha)$ multiplications, but only multiplies the degree of the ideal by $\alpha + 2$, rather than (as a loose upper bound) 4α. The term $\alpha + 2$ is peculiar in this context: it appears because of the particular way in which leading monomials can cancel each other, as explained in [BBL+24] and [Bri23, Chapter 11]. Indeed, we can verify experimentally that merely switching the signs in the nonlinear layer of Anemoi (so as to prevent leading monomial cancellation) can lead to a higher value for D_I.

- In the AIR model, we see that, much like other primitives, smaller values of α give a higher ALFOMs. Compared to Rescue, the primitive with the best asymptotic AIR ALFOMs (see Sec 4.1), Anemoi performs overall worse: not taking the round skips into account, we must compare $\log_2(\alpha+2)/(2\alpha)$ and $\log_2(\alpha)/\alpha$. As α grows, the factor $1/2$ that appears in Anemoi's ALFOMs implies that Anemoi should perform only half as well as Rescue. Concretely, if we take $\alpha = 3$ (the most interesting value in this context), we have that:

$$\begin{aligned} \eta_{AIR,Rescue}(\alpha = 3, b, r, k) &\underset{r\to\infty}{\sim} 0.528\,, \\ \eta_{AIR,Anemoi}(\alpha = 3, b, r, k) &\underset{r\to\infty}{\sim} 0.387\,. \end{aligned}$$

  This difference isn't really obvious in [BBC+23, Table 2], where Anemoi is listed to have less AIR constraints than Rescue in some instances with $\alpha = 3$ and a low number of branches (but it is clear that Rescue performs quite better for $\alpha = 5$, as expected). However, one should recall that the current version of Anemoi is not properly scaled with respect to D_I, which led to full-round PoSSo attacks in [BBL+24, BBB+25], particularly strong in instances with $\alpha = 3$ and a low number of branches. Assuming that Anemoi and Rescue are scaled in the same way, one should expect the ratio of their costs to be the same as the ratio of their ALFOMs.

5 Optimal Constructions: the Moirai

In this section, we study constructions that we prove or conjecture to be optimal for multiple variants of the ALFOMs metric. These round functions showcase how ALFOMs and their maximization also provide a useful guideline when designing round functions.

The three families of permutations we introduce are called the Moirai: Clotho, Lachesis, and Atropos.[8] To be clear, we are not making complete specifications for these constructions: we do not evaluate their security against other kinds of attacks, and we do not benchmark their performance, as that is outside of our interest. If one wished to do so, there is ample literature on (Extended) Generalized Feistel Networks [Nyb96, HR10, BMT14], which the Moirai belong to. Here we study the Moirai purely from the perspective of PoSSo cryptanalysis and ZK-friendliness, but fully fleshed-out designs may be different, in order to satisfy more complex constraints.

5.1 Defining the Moirai

Definition 6 (Moirai)**.** Given a prime number p, an even number of branches $2b$, a number of rounds r, a full rank matrix M, and a function $S : \mathbb{F}_p \to \mathbb{F}_p$, we define a Moirai as a permutation of $\mathbb{F}_p^{2b}$ equal to

$$\mathcal{A}_r \circ \mathcal{S} \circ \mathcal{A}_{r-1} \circ \mathcal{S} \circ \cdots \circ \mathcal{A}_1 \circ \mathcal{S} \circ \mathcal{A}_0,$$

[8] We followed the tradition of using names from Greek mythology. These are the three sisters that personify destiny.

Table 2: Qualitative comparison of the Moirai (MDS or Half-MDS layers).
$*$: Not as fast as Clotho, but faster than Lachesis.

Moirai	Computable D_I	S has high degree	S is injective	Can always have $d_{\max} \leq 4$	Fast evaluation
Clotho	✓	✗	✗	✓	✓
Lachesis	✓	✓	✓	✗	✗
Atropos	✗	✓	✗	✓	✗*

where, for $0 \leq j \leq r$, $\mathcal{A}_j$ denotes the multiplication by M followed by the addition of round constant $c_{i,j}$ on the i-th branch for $1 \leq i \leq b$, and $\mathcal{S}$ maps $(x_1, \ldots, x_b, y_1, \ldots, y_b)$ of $2b$ elements to $(x_1, \ldots, x_b, y_1 + S(x_1), \ldots, y_b + S(x_b))$.

This overall structure is summarized in Figure 1a. We define three particular subfamilies of the Moirai:

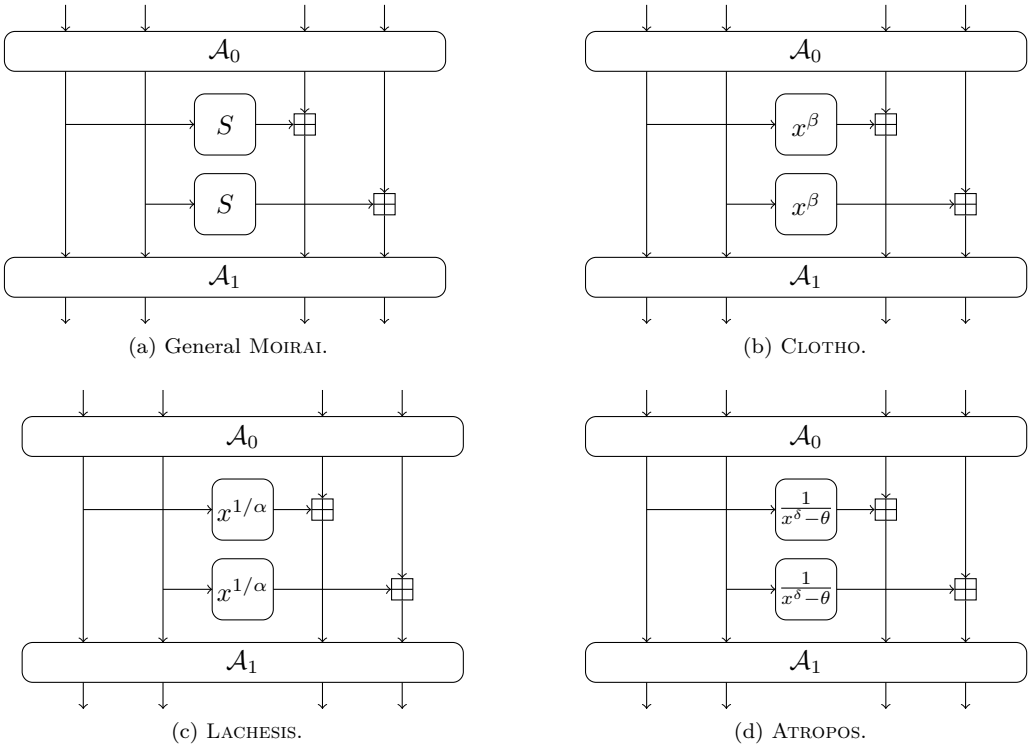
1. Clotho, for which $S : x \mapsto x^\beta$ for any $\beta \geq 2$, not necessarily coprime to $p-1$.
2. Lachesis, for which $S : x \mapsto x^{1/\alpha}$ for some α coprime to $p-1$.
3. Atropos, the ones for which $S : x \mapsto (x^\delta - \theta)^{-1}$, where $\delta \geq 2$ **must not be** coprime to $p-1$ (such as all powers of 2) and θ is not a δ-th power in $\mathbb{F}_p$. Such a value exists if and only if δ is not coprime with $p-1$.

Different Linear Layers. By using particular structures for the matrix $\mathbf{M}$, we obtain Moirai variants that are essentially Feistel networks (up to a renumbering of their input words). More generally, we will focus on two different structures for this matrix corresponding to the following two cases.

SPN case. We say that a Moirai *has MDS (affine) layers* when the matrix $\mathbf{M}$ itself is MDS. For a $2b \times 2b$ matrix, this means that for every nonzero vector $v \in \mathbb{F}_p^{2b}$, the sum of the number of nonzero coefficients of v and $\mathbf{M} \cdot v$ is greater than or equal to $2b+1$.

Feistel Network case. We say that it has Half-MDS (affine) layers if it is defined using two MDS matrices $\mathbf{M} \in \mathbb{F}_p^{2b \times 2b}$ and $\mathbf{M}' \in \mathbb{F}_p^{b \times b}$ as follows. The affine layer $\mathcal{A}_j$ for $0 < j < r$ is written $\mathcal{A}_j : \mathbf{v}_L \parallel \mathbf{v}_R \mapsto (\mathbf{v}_R \parallel \mathbf{M}' \times \mathbf{v}_L) + \mathbf{c_j}$, where $\mathbf{v}_L$ and $\mathbf{v}_R$ are vectors of size b, $\mathbf{c_j} = (c_{1,j}, \ldots, c_{2b,j})$, while the first and last affine layers (i.e. for $j \in \{0, r\}$) are such that $\mathcal{A}_j = \mathbf{M} + \mathbf{c_j}$.

Properties of Specific Moirai. Giving a full specification of the Moirai falls outside of the scope of this paper, and while it may be an interesting future work, the point of this section is only to study them as "ALFOMs-efficient" primitives. In the following subsections, we will show that the Clotho family can reach the optimal ALFOMs for both R1CS and AIR for CICO-b (but not at the same time), where the total number of branches is $2b$. We also show that Lachesis can be AIR-optimal, but not R1CS-optimal. Lastly, we show that Atropos has some practical advantages over Lachesis, although computing precisely the value of D_I is unfeasible at this stage. In some cases where we conjecture the Bézout bound on D_I to be tight, we show that Atropos can be AIR-optimal, but not R1CS-optimal, like Lachesis. We consider Lachesis and Atropos as interesting alternatives whose nonlinear layers do not have a low degree. We show in particular that their ALFOMs can be arbitrarily close to the optimal value, as the security level tends to infinity.



(a) General Moirai. (b) Clotho.

(c) Lachesis. (d) Atropos.

Figure 1: First three layers of a Moirai.

5.2 Analysis of Clotho

Recall (Definition 6) that Clotho features a nonlinear layer that applies the $(x, y) \mapsto (x, y + x^\beta)$ in parallel, for some nonzero integer β.

Proposition 5 (D_I of Clotho)**.** *Given an instance of Clotho over $2b$ branches with either MDS or half-MDS layers, consider the CICO-b problem for $k \leq b$. It can be solved with a polynomial system of $b + rb$ equations and variables whose ideal degree is*

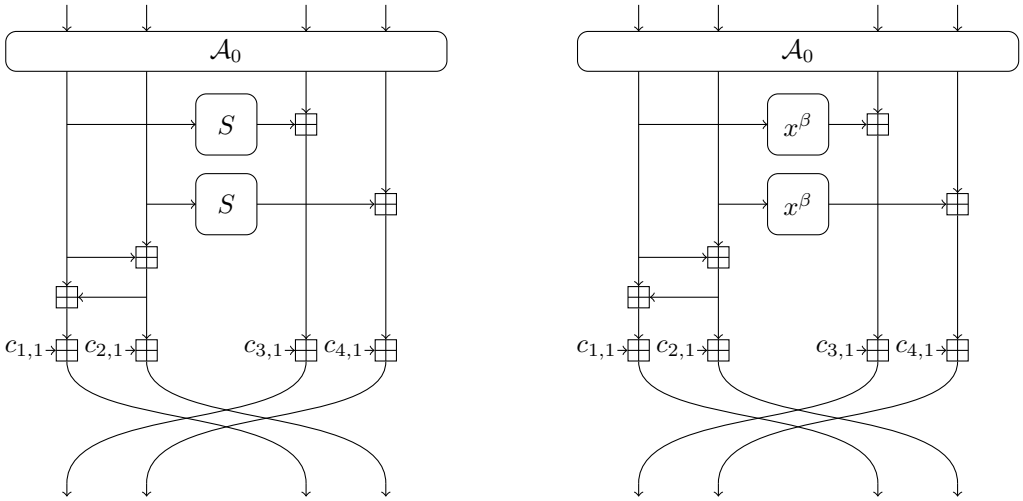
$$D_I = \beta^{b \cdot r}. \tag{3}$$

Proof. We can compute D_I exactly by constructing a polynomial model that solves the CICO-b problem, and that is already a Gröbner basis in grevlex order. To this end, we follow the Freelunch [BBL+24] method. Here we do the simplest case, where the Clotho instance we consider has half-MDS affine layers. The case where Clotho has MDS affine layers also works, but it is a bit more technical, and we describe the proof in Appendix C. Denote $\mathbf{M}$ and $\mathbf{M}'$ as the matrices that define the MDS and half-MDS layers respectively. First, we represent the input of the CICO-b problem as the sum $x_{1,1}\mathbf{v}_{1,0} + \cdots + x_{b,1}\mathbf{v}_{b,0}$. Each vector $\mathbf{v}_{i,0} = (v_{i,0,1}, \ldots, v_{i,0,b}, 0, \ldots, 0) \in \mathbb{F}_p^{2b}$ is defined by a linear system of the form

$$\mathbf{M} \cdot \mathbf{v}_{i,0} = (\underbrace{0, \ldots, 0, 1}_{i}, \underbrace{0 \ldots, 0}_{b-i}, \underbrace{*, \ldots, *}_{b})$$

where 1 is at the i-th coordinate[9], i.e. $\mathbf{M} \cdot \mathbf{v_{i,0}}$ is equal to 0 on all left-side coordinates except at i where it is equal to 1. Such a $\mathbf{v}_{i,0}$ always exists because $\mathbf{M}$ is MDS. Now,

[9]Here, the index of the first coordinate is 1.



(a) General Moirai. (b) Clotho.

Figure 2: First three layers of a Moirai with half-MDS layers. $\mathbf{M}'$ is instantiated as the MDS operation $(x_0, x_1) \mapsto (2x_0 + x_1, x_0 + x_1)$.

we proceed by induction. Let $1 \leq j \leq r$ and $1 \leq i \leq b$. Suppose that, at the input of the j-th $\mathcal{S}$ layer, the i-th branch is a degree-1 polynomial denoted $L_{i,j}$ in variables $\{x_{i,j-1}\} \cup \{x_{i',j'} : 1 \leq i' \leq b, 0 \leq j' < j-1\}$. As we have seen, this is true for $j = 1$. We then introduce the variable $x_{i,j}$ for $1 \leq i \leq b$ at the output of the S function of the i-th branch. Note that, in the following round, due to the structure of the half-MDS affine layers, $x_{i,j}$ will still only be on the i-th branch, which validates our induction hypothesis. It verifies the equation $x_{i,j} = L_{i,j}(\{x_{i,j-1}\} \cup \{x_{i',j'} : 1 \leq i' \leq b, 0 \leq j' < j-1\})^\beta$. The leading monomial of this equation is $x_{i,j-1}^\beta$, assuming that we use the grevlex order with a lexicographical order such that, if $j_1 < j_2$, then $x_{i_1,j_1} < x_{i_2,j_2}$ for all i_1, i_2, j_1, j_2. By induction, we then fully define the model. The last b equations are affine and can be reduced so that their leading monomials are precisely $\{x_{i,r} : 1 \leq i \leq b\}$. We know that this is always possible because the last layer is an MDS layer, and the $\{x_{i,r} : 1 \leq i \leq b\}$ variables are diffused to all of the output branches. In doing so, we have ensured that the set of leading monomials of the model with the appropriate grevlex order is exactly $\{x_{i,j}^2 : 1 \leq i \leq b, 0 \leq j < r\} \cup \{x_{i,r} : 1 \leq i \leq b\}$. They are coprime, therefore they define a Gröbner basis (see [BBL+24, Prop. 4]). Its ideal degree is the product of the degrees of the leading monomials, and we deduce D_I. □

Proposition 6 (Costs of Clotho)**.** *Given an instance of Clotho over $2b$ branches, it is possible to encode it using R1CS or AIR with the following costs:*

$$C_{R1CS}(\beta, b, r) = \ell(\beta) \cdot b \cdot r \ , \quad C_{AIR}(\beta, b, r) = \beta \cdot 2b \cdot \left\lceil \frac{r}{2} \right\rceil \ .$$

Proof. The R1CS cost is immediate: each of the $b \cdot r$ Sboxes adds $\ell(\beta)$ quadratic constraints. For the AIR complexity, we adopt a similar modeling strategy as in Rescue [AAB+20], since $\mathcal{S}$ and its inverse have degree β. We introduce new variables every two rounds, such that the equations are defined by $\mathcal{S} \circ \mathcal{A}_{2i-2}(\mathbf{x}_i) = \mathcal{A}_{2i-1}^{-1} \circ \mathcal{S}^{-1}(\mathbf{x}_{i+1})$, for $0 \leq i \leq (\lceil r/2 \rceil - 2)$. For the last round, the equations are

$$\mathcal{A}_{2\lceil r/2 \rceil - 1}\mathcal{S} \circ \mathcal{A}_{2\lceil r/2 \rceil - 2}(\mathbf{x}_{\lceil r/2 \rceil - 1}) = \begin{cases} \mathcal{S}^{-1} \circ \mathcal{A}_r^{-1}(\mathbf{x}_{r/2}) & \text{if } r \text{ is even,} \\ \mathbf{x}_{(r+1)/2} & \text{if } r \text{ is odd,} \end{cases}$$

where $\mathbf{x}_0$ is the input and $\mathbf{x}_{\lceil r/2 \rceil}$ is the output. Altogether, we have $d_{\max} = \beta$, $w = 2b$ and $t = \lceil r/2 \rceil$, which we multiply to get C_{AIR}. □

Thus, using the models of Propositions 5 and 6, we can compute Clotho's ALFOMs.

Corollary 1. *For an instance of Clotho with $2b$ branches, $k = b$, and whose affine layers are MDS or half-MDS, the ALFOMs are*

$$\eta_{R1CS}(\beta, b, r) = \frac{\log_2(\beta)}{\ell(\beta)} \ , \quad \eta_{AIR}(\beta, b, r) = \begin{cases} \dfrac{\log_2(\beta)}{\beta} & \text{if } r \text{ even,} \\ \dfrac{\log_2(\beta)}{\beta}\left(1 - \dfrac{1}{r}\right) & \text{otherwise.} \end{cases}$$

Proof of Corollary 1. The formulas can be deduced from Propositions 5 and 6. For $m \geq 1$ and $\beta = 2^m$, we have that $\log_2(\beta) = \ell(\beta) = m$, which implies that $\eta_{R1CS}(2^m, b, r) = 1$. For AIR, we have that

$$\eta_{AIR}(\beta, b, r) = \frac{\log_2(\beta) \cdot b \cdot r}{\beta \cdot 2b \cdot \lceil r/2 \rceil},$$

and $r/(2\lceil r/2 \rceil)$ is equal to 1 if r is even, and $1 - 1/r$ if r is odd. □

Regarding optimality, we know from Propositions 3 and 4 that η_{R1CS} and η_{AIR} are respectively upper-bounded by 1 and $\log_2(3)/3$. For R1CS, the values of β that maximize η_{R1CS} are the powers of 2 (see Proposition 11), and if additionally r is even, then Clotho is R1CS-optimal, as for any $m \geq 1$, $\eta_{R1CS}(2^m, b, r) = 1$. For AIR, the value of β that maximizes η_{AIR} is 3, for which Clotho is AIR-optimal. The second-best are 2 and 4, which also yield R1CS-optimal instances:

$$\eta_{AIR}(3, b, r) = \frac{\log_2(3)}{3} \approx 0.53, \quad \eta_{AIR}(2, b, r) = \eta_{AIR}(4, b, r) = 0.5 \ .$$

When $\gcd(\beta, p-1)$ is big. The $x \mapsto x^\beta$ function is not necessarily a permutation. Depending on the factorization of $p-1$, it may be, at worst, a β-to-1 function. If p is small and β is big, this may lead to some weaknesses against non-PoSSo-based cryptanalysis. While outside the scope of this paper, it is still worth noting.

5.3 Analysis of Lachesis

Clotho, while R1CS-optimal in some cases, has the potentially unfortunate property of being of overall low degree (compared to random permutations of $\mathbb{F}_p^n$). Indeed, the quadratic permutations described currently in the literature [GOPS22, GGOP23] are all Feistel or Lai-Massey constructions, which have low-degree inverse. This can enable up some attack vectors, as evidenced by a recent paper [GKR25] which exploits a similar property in Neptune.

This observation motivated our design of Lachesis, which uses $x \mapsto x^{1/\alpha}$ as its nonlinear layer. The inverse being of low degree, it is easy to derive low-degree equations for both AIR and R1CS. However, since the degree of the evaluation is very high, the permutation itself will quickly reach a very high degree.

The use of the inverse of low-degree monomial is not new, it was already introduced by the designers of Rescue. However, this approach has a cost both in terms of R1CS constraints, and in terms of field dependence. Indeed, a high degree function enabling a low-degree verification ultimately relies on the inverse of low-degree monomials (in Rescue, Anemoi and Griffin), or on the inverse function (Vision). However, neither can allow a good ALFOM for AIR since the addition chains for the exponent are not of optimal length. In light of this observation, we make the following conjecture:

Conjecture 1. *Any permutation that is R1CS-optimal, that is, for which $\eta_{R1CS} = 1$ for some set of parameters, must be a composition of permutations that are of low degree, and whose inverse have a low degree. In other words, it is impossible to have both R1CS-optimality and a high-degree round function.*

Another key difference between the use of nonlinear functions of low degree and nonlinear functions with low-degree verification lies in the cryptanalyses they allow. In the case of a low-degree evaluation, it is possible to directly solve a univariate system obtained by simply applying the permutation to a symbolic variable. This is known to have a time complexity quasi-linear in D_I. On the other hand, forcing the introduction of intermediate variables may also a priori force cryptanalysts to use algorithms with a complexity proportional to D_I^ω or higher. Still, recent breakthroughs in Gröbner Basis and resultant-based attacks [BBL+24, YZY+24, BBB+25] achieve a complexity which is lower than $\widetilde{\mathcal{O}}(D_I^\omega)$ due to dedicated computer algebra techniques. Thus, in the remainder of this paper, we assume that all primitives are simply scaled so that $D_I \geq 2^\lambda$.

Proposition 7 (D_I of Lachesis)**.** *Given an instance of Lachesis over $2b$ branches, consider the CICO-b problem for $k \leq b$. It can be solved with a system of $b + rb$ equations and variables whose ideal degree is: $D_I = \alpha^{b \cdot r}$.*

Proof. To solve CICO-b on Lachesis, we define variables $x_{1,0}, \ldots, x_{b,0}$ at the input. Then, we add an intermediate variable at the output of each $x \mapsto x^{1/\alpha}$ function. We obtain, for each $1 \leq i \leq b$ and $1 \leq j \leq r$, an equation of the form $x_{i,j}^\alpha = L_{i,j}(x_{i',j'} : 1 \leq i' \leq b, 1 \leq j' < j)$, whose leading monomial is $x_{i,j}^\alpha$ in grevlex order. Finally, we get b affine equations at the output, which we can apply Gaussian reduction to, and in a lexicographical order where $x_{i,0} > x_{i',j'}$ for all $1 \leq i, i' \leq b$ and $1 \leq j' \leq r$, we get $x_{i,0}$ as leading monomials of the system for all i. Thus we have a system of $b + b \cdot r$ equations and variables, where the leading monomials are pairwise coprime, and so the degree of the ideal is equal to the product of their degrees. □

Proposition 8 (Costs of Lachesis)**.** *Given an instance of Lachesis over $2b$ branches, it is possible to encode it to an R1CS and an AIR with the following costs:*

$$C_{R1CS}(\alpha, b, r) = \ell(\alpha) \cdot b \cdot r \ , \quad C_{AIR}(\alpha, b, r) = \alpha \cdot 2b \cdot \left\lceil \frac{r}{2} \right\rceil \ .$$

Proof. For R1CS, the number of $x \mapsto x^{1/\alpha}$ functions is $b \cdot r$, and each one requires $\ell(\alpha)$ constraints. For AIR, we define a model with the following method: let us define the same variables as for the AIR model of Clotho (see the proof of 6). Let $0 \leq j \leq (\lceil r/2 \rceil - 2)$. We can define $2b$ independent equations of degree α between the variables of $\mathbf{x}_j$ and those of $\mathbf{x}_{j+1}$ with the following approach: consider part of the permutation $\mathcal{S} \circ \mathcal{A}_{j+1} \circ \mathcal{S}$, and let y_i and z_i for $1 \leq i \leq 2b$ be variables at the input and output it: they are degree-1 expressions in the variables of $\mathbf{x}_j$ and those of $\mathbf{x}_{j+1}$ respectively. Let t_i be the output of the $2i$-th branch after the first $\mathcal{S}$ layer, so that $t_i = S(y_{2i-1}) + y_{2i}$. We have that, for every $1 \leq i \leq b$, $z_{2i-1} = L_i(y_{2i'-1}, t_{i'} : 1 \leq i' \leq b)$, where L_i is an affine function. Now, we can rearrange these b affine equations in $3b$ variables, in order to express the $t_{i'}$ variables as affine functions of the others: for every $1 \leq i \leq b$, there exists an affine polynomials R_i such that $t_i = R_i(y_{2i'-1}, z_{2i'-1} : 1 \leq i' \leq b)$. By substituting t_i with this affine expression, we can derive an equation of degree α: $y_{2_i-1} = (R_i(y_{2i'-1}, z_{2i'-1} : 1 \leq i' \leq b) - y_{2i})^\alpha$. Symmetrically, by applying the same method on the $\mathcal{S}$ situated on the "other side" of the $\mathcal{A}_{j+1}$ layer, we also get b equations of the shape $z_{2_i-1} = (R'_i(z_{2i'-1}, y_{2i'-1} : 1 \leq i' \leq b) - z_{2i})^\alpha$. Finally, by replacing the y and z variables by their affine expressions in the $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$ variables respectively, we get the final $2b$ equations of degree α that are part of our AIR. When r is odd, for the final round, a slightly different (but simpler) approach lets us represent the final round with $2b$ equations of degree α between the variables of

$\mathbf{x}_{r-1}$ and $\mathbf{x}_r$, much like with Clotho. Finally, we get an AIR with $d_{\max} = \alpha$, $w = 2b$ and $t = \lceil r/2 \rceil$. □

Corollary 2 (ALFOMs of Lachesis)**.** *Given the models of Propositions 7 and 8, we have that, for an instance of Lachesis with $2b$ branches and $k = b$, its ALFOMs are*

$$\begin{aligned} \eta_{R1CS}(\alpha, b, r) &= \frac{\log_2(\alpha)}{\ell(\alpha)}, \\ \eta_{AIR}(\alpha, b, r) &= \begin{cases} \dfrac{\log_2(\alpha)}{\alpha} & \text{if } r \text{ is even,} \\ \dfrac{\log_2(\alpha)}{\alpha}\left(1 - \dfrac{1}{r}\right) & \text{if } r \text{ is odd.} \end{cases} \end{aligned}$$

For R1CS, fix $m \geq 2$. The value of α that maximizes η_{R1CS}, under the condition that each round has at most $b \cdot m$ quadratic constraints, is $2^{m-1} + 1$ (see Proposition 11 from Appendix B) when it is coprime with $p - 1$. For this value, and for r even,

$$\eta_{R1CS}(2^{m-1} + 1, b, r) = \frac{\log_2(2^{m-1} + 1)}{m} \underset{m \to \infty}{\sim} 1.$$

For AIR, the value of α that maximizes η_{AIR} is 3, when it is coprime with $p - 1$. For this value, and r even, $\eta_{AIR}(3, b, r) = \log_2(3)/3 \approx 0.53$, in which case Lachesis is AIR-optimal.

5.4 Analysis of Atropos

Atropos is the last Moirai we define. It uses the high-degree $x \mapsto 1/(x^\delta - \theta)$ functions, where θ is not a δ-th power in $\mathbb{F}_p$ (thus, δ must not be coprime to $p - 1$). The inverse function makes Atropos a bit harder to analyze, and it seems like this construction is redundant compared to Lachesis, in that it also has high-degree Sboxes at the cost of non-optimality with respect to R1CS. However, we identify some notable advantages compared to Lachesis and primitives that use $x \mapsto x^{1/\alpha}$ in general:

- For R1CS, we can always choose δ as a multiple of 2, so we can always have a cost of m multiplications to multiply D_I by $2^{m-1} + 1$. This mimics the best choice of $\alpha = 2^{m-1} + 1$ in Lachesis. However, in Lachesis (and similarly, primitives like Rescue [AAB+20]), we may not be able to choose the most interesting low values for α: $\alpha = 3$, $\alpha = 5$, or $\alpha = 9$—the highest exponents to require respectively 2, 3 and 4 multiplications. In fact, none of these can be chosen for $p = 2^{64} - 2^{32} + 1$, which is why XHash [ABK+23] has to resort to $\alpha = 7$.

- For AIR, similarly, the most interesting value, $d_{\max} = 3$, can always be chosen. That is not the case for Lachesis, Rescue, XHash, Anemoi or Griffin for the following widely used primes: $2^{64} - 2^{32} + 1$ [ABK+23], the Mersenne prime $2^{31} - 1$, and the primes of the curves BLS12-381 and BN-254 [BBC+23].

- In $\mathbb{F}_p$, with modular arithmetic, the inverse function can be computed more efficiently than high-degree power maps, using the extended Euclidean algorithm[10]. Additionally, [BZ10, Sec. 2.5.1] describes an efficient algorithm to compute multiple modular inversions modulo the same number at the same time.

[10] Concretely, we can measure the difference in Sagemath: for the p of BLS12-381, on a laptop, computing a million `inverse_mod(_,p)` takes about 3.0s, whereas a million `pow(_,alphainv,p)`, with `alphainv` the precomputed modular inverse of 5, takes about 24.0s. For $p = 2^{31} - 1$, the difference is smaller: 1.0s for the former, 2.0s for the latter.

On the inverse function. The "inverse" function is a bit tricky to work with. Indeed, we call it like that (and the authors of Vision even write it x^{-1}) even though 0 has no inverse. Implicitly, we just map 0 to 0 to complete the function, which actually yields the function $x \mapsto x^{p-2}$. This potentially induces a gap between the constraint system and cryptanalysis: given an equation $y = x^{p-2}$, the constraint system must model it as $yx^2 = x$, instead of $yx = 1$, to take into account the case where $x = 0$. On the other hand, a cryptanalyst may assume that the inputs of the inverse Sboxes are never equal to 0 (which becomes more likely as p grows) and solve for equations of degree 2 instead of 3. This issue can be resolved by making sure that inverse Sboxes never receive 0 as an input. For instance, we can first map the input x to $x^2 - \theta$ (where θ is not a square in the base field), and apply the inverse to the result. This idea led us to define Atropos.

Proposition 9 (D_I of Atropos)**.** *Given an instance of Atropos over $2b$ branches, the CICO-b problem for $k \leq b$ can be solved with a polynomial system of $b + rb$ equations and variables whose ideal degree verifies: $D_I \leq (\delta + 1)^{b \cdot r}$.*

Proof. Since we are not explicitly building a Gröbner basis, the proof is quite simple. We define the input state for the CICO-b as $(x_{1,0}, \ldots, x_{b,0}, 0, \ldots, 0)$. Let $1 \leq i \leq b$ and $1 \leq j \leq r$. We define the variable $x_{i,j}$ as the output of the i-th $x \mapsto 1/(x^\delta - \theta)$ function of the j-th $\mathcal{S}$ layer. Inductively, the input of this function will be a degree-1 polynomial in all variables defined over all previous rounds, which we call $L_{i,j}$. Then, the corresponding equation has the shape

$$x_{i,j} \cdot \left(L_{i,j}(x_{1,0}, \ldots, x_{b,0}, \ldots, x_{1,j-1}, \ldots, x_{b,j-1})^\delta - \theta\right) = 1.$$

Each of these $b \cdot r$ equations has total degree $\delta + 1$. Finally, b degree-1 equations that model the output of the CICO-b problem complete the system. Therefore, we deduce the result from the Bézout bound: D_I is at most the product of the total degrees of all of the equations. □

Experimental verifications. The results of our experiments are summarized in Appendix E (Table 4 in particular). We notice a big difference between MDS layers and non-MDS layers: the Bézout bound is tight in all of the instances we verified with MDS layers, but those that have non-MDS layers have lower D_I than expected starting at $r = 3$. Nevertheless, we conjecture that in the MDS layers case, the Bézout bound is tight in general for the model we described. Verifying that conjecture and/or explicitly describing a Gröbner basis for this system for various kinds of affine layers remain open questions, and we leave them for future work. For the remainder of this section (in particular, to compute Atropos's ALFOMs), we will assume that the Bézout bound is, in fact, tight, when using MDS layers.

Proposition 10 (Costs of Atropos)**.** *An instance of Atropos over $2b$ branches, can be encoded into an R1CS or an AIR with the following costs:*

$$C_{R1CS}(\delta, b, r) = (\ell(\delta) + 1) \cdot b \cdot r \ , \quad C_{AIR}(\delta, b, r) = (\delta + 1) \cdot 2b \cdot \left\lceil \frac{r}{2} \right\rceil \ .$$

Proof. For R1CS, the input x_1 and the output x_2 of the $x \mapsto (x^\delta - \theta)^{-1}$ function are linked by the equation $x_2 \cdot (x_1^\delta - 1) = 1$. Furthermore, x_1^δ can be computed from x_1 using $\ell(\delta)$ quadratic constraints, and one additional quadratic constraint is used to multiply $x_1^\delta - \theta$ by x_2. For AIR, we can reuse the same method that we described for Lachesis (see the proof of Proposition 8). Using the same notations, the only difference is that, instead of the equation $y_{2i-1} = (R_i(y_{2i'-1}, z_{2i'-1}) - y_{2i})^\alpha$ of degree α (and its symmetrical equivalent), we have instead the equation $(y_{2i-1}^\delta - \theta)(R_i(y_{2i'-1}, z_{2i'-1}) - y_{2i}) = 1$, which is of degree $\delta + 1$. Thus, we get that $d_{\max} = \delta + 1$, $w = 2b$ and $t = \lceil r/2 \rceil$. □

Corollary 3 (ALFOMs of Atropos)**.** *Consider an instance of Atropos with $2b$ branches and $k = b$. Suppose that the Bézout bound for D_I, given in Prop. 9, is an equality. Then, Atropos's ALFOMs are*

$$\eta_{R1CS}(\delta, b, r) = \frac{\log_2(\delta+1)}{\ell(\delta)+1},$$
$$\eta_{AIR}(\delta, b, r) = \begin{cases} \dfrac{\log_2(\delta+1)}{\delta+1} & \text{if } r \text{ is even,} \\ \dfrac{\log_2(\delta+1)}{\delta+1}\left(1-\dfrac{1}{r}\right) & \text{if } r \text{ is odd.} \end{cases}$$

For R1CS, like for Lachesis, given a fixed number $b \cdot m$ of quadratic constraints per round, the value of δ that maximizes η_{R1CS} is 2^{m-1}. For this value, and for r even,

$$\eta_{R1CS}(2^{m-1}, b, r) = \frac{\log_2(2^{m-1}+1)}{m} \underset{m\to\infty}{\sim} 1.$$

For AIR, the value of δ that maximizes η_{AIR} is 2. For this value, when r is even, we have $\eta_{AIR}(2, b, r) = \dfrac{\log_2(3)}{3} \approx 0.53$, meaning that Atropos is AIR-optimal.

5.5 Experimental results

The results are summarized in Table 3. Note that we always use the best known round skips to compute D_I, but there may exist better round skips that are not known yet. That is why we focus on CICO-b: as the attacker has much less freedom for this problem, we are more confident that our analysis does not miss any significant round skip.

Table 3: Costs with r computed so that $D_I \geq 2^{128}$. Next to the cost, the relevant ALFOMs is shown in parenthesis. The best costs per column are in bold. All Bézout bounds are assumed to be tight.

		R1CS cost				AIR cost			
		$b=2$	$b=4$	$b=6$	$b=8$	$b=2$	$b=4$	$b=6$	$b=8$
Clotho	$\beta=2$	**128 (1.0)**	**128 (1.0)**	**129 (1.0)**	**128 (1.0)**	256 (0.50)	256 (0.50)	264 (0.49)	**256 (0.50)**
	$\beta=3$	162 (0.79)	164 (0.79)	162 (0.79)	168 (0.79)	**246 (0.52)**	**252 (0.52)**	**252 (0.51)**	264 (0.50)
Lachesis	$\alpha=3$	162 (0.79)	164 (0.79)	162 (0.79)	168 (0.79)	**246 (0.52)**	**252 (0.52)**	**252 (0.51)**	264 (0.50)
	$\alpha=5$	168 (0.77)	168 (0.77)	171 (0.77)	168 (0.77)	280 (0.46)	280 (0.46)	300 (0.44)	280 (0.46)
Atropos	$\delta=2$	162 (0.79)	164 (0.79)	162 (0.79)	168 (0.79)	**246 (0.52)**	**252 (0.52)**	**252 (0.51)**	264 (0.50)
Rescue-Prime	$\alpha=3$	336 (0.39)	352 (0.38)	360 (0.37)	384 (0.36)	252 (0.52)	264 (0.50)	270 (0.49)	288 (0.48)
	$\alpha=5$	348 (0.37)	360 (0.36)	396 (0.35)	384 (0.34)	290 (0.45)	300 (0.43)	330 (0.42)	320 (0.41)
Anemoi	$\alpha=3$	224 (0.58)	224 (0.58)	228 (0.58)	224 (0.58)	336 (0.39)	336 (0.39)	342 (0.39)	336 (0.39)
	$\alpha=5$	230 (0.56)	230 (0.56)	240 (0.56)	240 (0.56)	460 (0.28)	460 (0.28)	480 (0.28)	480 (0.28)
Poseidon	$\alpha=3$	180 (0.71)	200 (0.64)	220 (0.58)	240 (0.53)	270 (0.48)	300 (0.43)	342 (0.38)	360 (0.36)
	$\alpha=5$	195 (0.67)	225 (0.58)	255 (0.51)	285 (0.46)	330 (0.39)	380 (0.34)	450 (0.29)	480 (0.27)

As expected, Clotho with $\beta = 2$ has, by far, the lowest R1CS costs (all very close to $\lambda = 128$). It is the only R1CS-optimal primitive, whereas all other primitives in the table have an η_{R1CS} of less than 0.8. Note how two primitives with the same ALFOMs may not have the same costs: there is a small discrepancy due to the fact that D_I is not necessarily perfectly equal to 2^λ, which adds a small overhead. Indeed, when an ALFOM has the maximum possible value, it means that that value of D_I cannot be reached with a lower cost. So, a round function is "truly optimal" for a security parameter λ when its ALFOM is the maximum it can be AND $D_I = 2^\lambda$. Notably, the difference between D_I and 2^λ tends to grow with the number of branches b due to this "border effect", which is why we can observe that on any given row, increasing b tends to increase the cost and decrease the ALFOM.

For R1CS, Clotho, Lachesis and Atropos reach an ALFOM of either 1, $\log_2(3)/2 \approx 0.79$ or $\log_2(5)/3 \approx 0.77$. This is higher than all the other ones: Rescue-Prime has an ALFOM of approximately $\log_2(\alpha)/(2 \cdot \ell(\alpha))$, so it performs about half as well as a Moirai with the "same α". The higher level explanation for that is that each round of Rescue has two layers full of either x^α or $x^{1/\alpha}$ Sboxes, which, overall, contributes as much as two rounds of a Moirai with the "same α" to D_I, even though each nonlinear layer of a Moirai has half as many Sboxes. Anemoi performs worse than the Moirai because each Flystel contributes only a factor $\alpha + 2$ to D_I, even though it has two x^2 and one $x^{1/\alpha}$ Sboxes. Comparatively, the same Sboxes in a more adequate design could contribute a factor $4 \cdot \alpha$. Finally, when b is low, Poseidon gets close to the ALFOMs of the Moirai with the "same α": it cannot get better than the Moirai because it does not use any x^2 Sbox, and it performs marginally worse than the Moirai because the Sboxes on the full rounds don't all contribute a factor α to D_I. It is also for that reason that Poseidon seems to perform worse and worse as b grows, both in R1CS and AIR.

For AIR, the best scores are more distributed: the Moirai and Rescue-Prime almost reach the optimal ALFOM of $\log_2(3)/3 \approx 0.53$ for these low values of α, β and δ. For these parameters they do not quite reach it: Rescue-Prime has round skips, and the number of rounds for the Moirai for $\alpha = 3$ is always an odd integer, thus the "two-round folding" can't be performed perfectly. The Moirai could, however, reach an optimal ALFOM with different security parameters. Anemoi is the worst one for AIR, never reaching an ALFOM above 0.4, and Poseidon performs pretty well for the lowest α and b, though it is never better than Rescue-Prime or a Moirai.

6 Conclusion

The rigid framework provided by the ALFOMs is a powerful tool. We can now compare most primitives discussed in this paper in a fair way that takes into account both their security and their performance. Given a security parameter λ, a number of branches $2b$, considering CICO-b (i.e. $k = b$), and all primitive-wise parameters aside from r, define r as $r = \min\{r : D_I(r, k = b) \geq 2^\lambda\}$. Naturally, this definition of r is much too simple for a full specification: marginal rounds should always be added, and in some cases designers may want to argue in favor of a more aggressive criterion on D_I, such as $D_I^2 \geq 2^\lambda$. Nevertheless, we maintain that the resulting comparison remains relevant.

To the best of our knowledge, the case of R1CS and AIR-friendly hash functions is the first where it is possible to identify such a clear description of the tradeoff between performance and security. Thanks to the ALFOMs, it is now possible to make absolute statements about the optimality (or lack thereof) of a given round function, and to make fair comparison of the round functions of primitives that encompass both their performance and the security they offer.

The full picture is of course a bit more complicated because of round-skipping, but we believe that the ALFOMs are the proper way to compare ZK-friendly hash functions in the case of R1CS or AIR arithmetizations.

Designing a similar figure of merit for Plonk is an interesting research direction, but this arithmetization gives more freedom to implementers. Paradoxically, this greater freedom is a hindrance to our approach, at least in its current shape.

Finally, we hope that the directions sketched by the Moirai will lead to the design of better AOPs in the future.

Acknowledgments

The work of Aurélien Boeuf and Léo Perrin was supported by the European Research Council (ERC, grant agreement no. 101041545 "ReSCALE"). The work of Aurélien Boeuf

was also supported in part by the French Agence Nationale de la Recherche through the SWAP project under Contract ANR-21-CE39-0012. We thank Augustin Bariant for some editorial remarks.

References

[AAB+20] Abdelrahaman Aly, Tomer Ashur, Eli Ben-Sasson, Siemen Dhooghe, and Alan Szepieniec. Design of symmetric-key primitives for advanced cryptographic protocols. *IACR Trans. Symm. Cryptol.*, 2020(3):1–45, 2020. doi:10.13154/tosc.v2020.i3.1-45.

[ABK+23] Tomer Ashur, Amit Singh Bhati, Al Kindi, Mohammad Mahzoun, and Léo Perrin. XHash: Efficient STARK-friendly hash function. Cryptology ePrint Archive, Report 2023/1045, 2023. URL: https://eprint.iacr.org/2023/1045.

[ACG+19] Martin R. Albrecht, Carlos Cid, Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, and Markus Schofnegger. Algebraic cryptanalysis of STARK-friendly designs: Application to MARVELlous and MiMC. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 371–397. Springer, Cham, December 2019. doi:10.1007/978-3-030-34618-8_13.

[AGR+16] Martin R. Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 191–219. Springer, Berlin, Heidelberg, December 2016. doi:10.1007/978-3-662-53887-6_7.

[AKM+22] Tomer Ashur, Al Kindi, Willi Meier, Alan Szepieniec, and Bobbin Threadbare. Rescue-prime optimized. Cryptology ePrint Archive, Report 2022/1577, 2022. URL: https://eprint.iacr.org/2022/1577.

[BBB+25] Augustin Bariant, Aurélien Boeuf, Pierre Briaud, Ma ël Hostettler, Morten Øygarden, and Håvard Raddum. Improved resultant attack against arithmetization-oriented primitives. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part V*, volume 16004 of *LNCS*, pages 335–367. Springer, Cham, August 2025. doi:10.1007/978-3-032-01901-1_11.

[BBC+23] Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, Robin Salen, Vesselin Velichkov, and Danny Willems. New design techniques for efficient arithmetization-oriented hash functions: Anemoi permutations and Jive compression mode. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 507–539. Springer, Cham, August 2023. doi:10.1007/978-3-031-38548-3_17.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. URL: https://eprint.iacr.org/2018/046.

[BBL+24] Augustin Bariant, Aurélien Boeuf, Axel Lemoine, Irati Manterola Ayala, Morten Øygarden, Léo Perrin, and Håvard Raddum. The algebraic FreeLunch: Efficient Gröbner basis attacks against arithmetization-oriented primitives.

In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part IV*, volume 14923 of *LNCS*, pages 139–173. Springer, Cham, August 2024. doi:10.1007/978-3-031-68385-5_5.

[BBLP22] Augustin Bariant, Clémence Bouvier, Gaëtan Leurent, and Léo Perrin. Algebraic attacks against some arithmetization-oriented primitives. *IACR Trans. Symm. Cryptol.*, 2022(3):73–101, 2022. doi:10.46586/tosc.v2022.i3.73-101.

[BCP23] Aurélien Boeuf, Anne Canteaut, and Léo Perrin. Propagation of subspaces in primitives with monomial sboxes: Applications to rescue and variants of the AES. *IACR Trans. Symm. Cryptol.*, 2023(4):270–298, 2023. doi:10.46586/tosc.v2023.i4.270-298.

[BDPVA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponge functions. Technical report, Keccak Team, 2011. URL: https://keccak.team/files/CSF-0.1.pdf.

[Ber71] Elwyn R. Berlekamp. Factoring polynomials over large finite fields. In Stanley R. Petrick, Jean E. Sammet, Robert G. Tobey, and Joel Moses, editors, *Proceedings of the second ACM symposium on Symbolic and algebraic manipulation, SYMSAC 1971, Los Angeles, California, USA, March 23-25, 1971*, page 223. ACM, 1971. doi:10.1145/800204.806290.

[BGL20] Eli Ben-Sasson, Lior Goldberg, and David Levit. STARK friendly hash – survey and recommendation. Cryptology ePrint Archive, Report 2020/948, 2020. URL: https://eprint.iacr.org/2020/948.

[BMT14] Thierry P. Berger, Marine Minier, and Gaël Thomas. Extended generalized Feistel networks using matrix representation. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, *SAC 2013*, volume 8282 of *LNCS*, pages 289–305. Springer, Berlin, Heidelberg, August 2014. doi:10.1007/978-3-662-43414-7_15.

[BND22] Jérémy Berthomieu, Vincent Neiger, and Mohab Safey El Din. Faster change of order algorithm for Gröbner bases under shape and stability assumptions. In Marc Moreno Maza and Lihong Zhi, editors, *ISSAC '22: International Symposium on Symbolic and Algebraic Computation, Villeneuve-d'Ascq, France, July 4 - 7, 2022*, pages 409–418. ACM, 2022. doi:10.1145/3476446.3535484.

[BP25] Antoine Bak and Léo Perrin. On the security of split-and-lookup-based ZK-friendly primitives. *IACR Transactions on Symmetric Cryptology*, 2025(2):87–123, Jun. 2025. URL: https://tosc.iacr.org/index.php/ToSC/article/view/12245, doi:10.46586/tosc.v2025.i2.87-123.

[Bri23] Pierre Briaud. *Cryptanalyse Algébrique de Schémas Post-Quantiques et Hypothèses Associées. (Algebraic Cryptanalysis of Post-Quantum Schemes and Related Assumptions)*. PhD thesis, Sorbonne University, France, 2023. URL: https://tel.archives-ouvertes.fr/tel-04358018.

[BZ10] Richard P. Brent and Paul Zimmermann. Modern computer arithmetic (version 0.5.1). *CoRR*, abs/1004.4710, 2010. URL: http://arxiv.org/abs/1004.4710.

[CLO25] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms*. Undergraduate Texts in Mathematics. Springer Cham, 2025. doi:10.1007/978-3-031-91841-4.

[CR25] Luca Campa and Arnab Roy. Groebner basis cryptanalysis of Anemoi. Cryptology ePrint Archive, Paper 2025/814, 2025. URL: https://eprint.iacr.org/2025/814, doi:10.1007/978-3-031-91107-1_11.

[CZ81] David G. Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Math. Comput.*, 36:587–592, 1981. doi:10.2307/2007663.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999. doi:10.1016/S0022-4049(99)00005-5.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 international symposium on symbolic and algebraic computation, ISSAC 2002, Lille, France, July 07–10, 2002.*, pages 75–83. New York, NY: ACM Press, 2002. doi:10.1145/780506.780516.

[FGLM93] Jean-Charles Faugère, Patrizia M. Gianni, Daniel Lazard, and Teo Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *J. Symb. Comput.*, 16(4):329–344, 1993. URL: https://doi.org/10.1006/jsco.1993.1051, doi:10.1006/JSCO.1993.1051.

[FM17] Jean-Charles Faugère and Chenqi Mou. Sparse FGLM algorithms. *J. Symb. Comput.*, 80:538–569, 2017. URL: https://doi.org/10.1016/j.jsc.2016.07.025, doi:10.1016/J.JSC.2016.07.025.

[FO90] Philippe Flajolet and Andrew M. Odlyzko. Random mapping statistics. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *EUROCRYPT'89*, volume 434 of *LNCS*, pages 329–354. Springer, Berlin, Heidelberg, April 1990. doi:10.1007/3-540-46885-4_34.

[GGOP23] Ginevra Giordani, Lorenzo Grassi, Silvia Onofri, and Marco Pedicini. Invertible quadratic non-linear functions over $\mathbb{F}_p^n$ via multiple local maps. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 151–176. Springer, Cham, July 2023. doi:10.1007/978-3-031-37679-5_7.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Berlin, Heidelberg, May 2013. doi:10.1007/978-3-642-38348-9_37.

[GHR+23] Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schofnegger, Roman Walch, and Qingju Wang. Horst meets fluid-SPN: Griffin for zero-knowledge applications. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 573–606. Springer, Cham, August 2023. doi:10.1007/978-3-031-38548-3_19.

[GKf+24] Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lü ftenegger, Christian Rechberger, Markus Schofnegger, and Roman Walch. Monolith: Circuit-friendly hash functions with new nonlinear layers for fast and constant-time implementations. *IACR Trans. Symm. Cryptol.*, 2024(3):44–83, 2024. doi:10.46586/tosc.v2024.i3.44-83.

[GKR+21] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. Poseidon: A new hash function for zero-knowledge proof systems. In Michael Bailey and Rachel Greenstadt, editors, *USENIX Security 2021*, pages 519–535. USENIX Association, August 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/grassi.

[GKR25] Lorenzo Grassi, Katharina Koschatko, and Christian Rechberger. Poseidon and Neptune: Gröbner basis cryptanalysis exploiting subspace trails. Cryptology ePrint Archive, Paper 2025/954, 2025. URL: https://eprint.iacr.org/2025/954.

[GKS23] Lorenzo Grassi, Dmitry Khovratovich, and Markus Schofnegger. Poseidon2: A faster version of the poseidon hash function. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 177–203. Springer, Cham, July 2023. doi:10.1007/978-3-031-37679-5_8.

[GOPS22] Lorenzo Grassi, Silvia Onofri, Marco Pedicini, and Luca Sozzi. Invertible quadratic non-linear layers for MPC-/FHE-/ZK -friendly schemes over $\mathbb{F}_p^n$: Application to Poseidon. *IACR Trans. Symm. Cryptol.*, 2022(3):20–72, 2022. doi:10.46586/tosc.v2022.i3.20-72.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. URL: https://eprint.iacr.org/2019/953.

[HR10] Viet Tung Hoang and Phillip Rogaway. On generalized Feistel networks. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 613–630. Springer, Berlin, Heidelberg, August 2010. doi:10.1007/978-3-642-14623-7_33.

[Knu98] Donald Ervin Knuth. *The art of computer programming, Volume II: Seminumerical Algorithms, 3rd Edition*. Addison-Wesley, 1998. URL: https://www.worldcat.org/oclc/312898417.

[MA23] Irati Manterola Ayala. STAP zoo. https://stap-zoo.com/, 2023.

[Moe73] Robert T. Moenck. Fast computation of GCDs. In Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard M. Karp, and H. Raymond Strong, editors, *Proceedings of the 5th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1973, Austin, Texas, USA*, pages 142–151. ACM, 1973. doi:10.1145/800125.804045.

[Nyb96] Kaisa Nyberg. Generalized Feistel networks. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 91–104. Springer, Berlin, Heidelberg, November 1996. doi:10.1007/BFb0034838.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013. doi:10.1109/SP.2013.47.

[SAD20] Alan Szepieniec, Tomer Ashur, and Siemen Dhooghe. Rescue-prime: a standard specification (SoK). Cryptology ePrint Archive, Report 2020/1143, 2020. URL: https://eprint.iacr.org/2020/1143.

[YZY+24] Hong-Sen Yang, Qun-Xiong Zheng, Jing Yang, Quan feng Liu, and Deng Tang. A new security evaluation method based on resultant for arithmetic-oriented algorithms. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part VII*, volume 15490 of *LNCS*, pages 457–489. Springer, Singapore, December 2024. doi:10.1007/978-981-96-0941-3_15.

A Round function figures

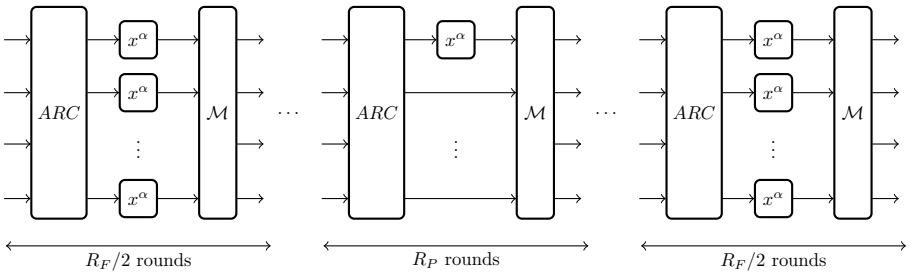


Figure 3: Poseidon round function.

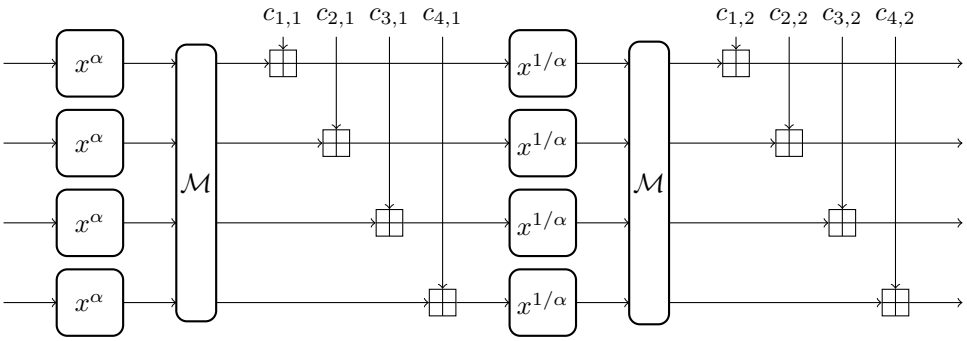


Figure 4: Rescue-Prime round function.

B On Addition Chains

When designing ZK-friendly primitives, designers often use permutations like $x \mapsto x^\alpha$ over finite fields, as they are easy to analyze and can be computed using few multiplications if α is not too big. Primitives like Rescue, Griffin, Arion and Anemoi also rely heavily on the functional inverse of $x \mapsto x^\alpha$. The designers mainly focus on the cases $\alpha = 3, 5$ or 7, as $x \mapsto x^2$ is never a permutation over any finite field. Those permutations require, respectively, $2, 3$ and 4 multiplications.

The minimal number of multiplications required to compute $x \mapsto x^\alpha$ is given by the length of the smallest addition chain of α (minus one). Here, we define addition chains and their properties:

Definition 7 (Addition Chain)**.** Let $n \geq 1$. An addition chain for n is an increasing sequence $k_0, \ldots, k_m$ such that

1. $k_0 = 1$.

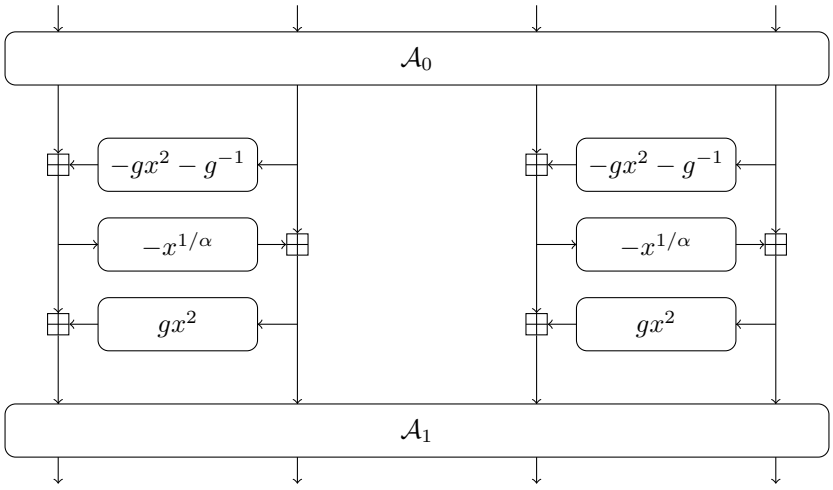


Figure 5: Anemoi's first layers.

2. For all $0 \leq i \leq m$, k_i is the sum of two previous numbers: there exist $0 \leq j_1, j_2 < i$ such that $k_i = k_{j_1} + k_{j_2}$.

3. $k_m = n$.

We denote the shortest length of all addition chains for n (that is, the smallest m such that $k_0, \ldots, k_m$ can be an addition chain) as $\ell(n)$. One important application of addition chains is that the smallest number of multiplications required to compute x^n from x is $\ell(n) - 1$.

Example 3 (Addition Chains)**.** An addition chain for 5 could be $(1, 2, 3, 4, 5)$. Indeed, $2 = 1 + 1$, $3 = 2 + 1$, $4 = 3 + 1$ and $5 = 4 + 1$.

However, $(1, 2, 4, 5)$ and $(1, 2, 3, 5)$ are shorter (and, in fact, the shortest) addition chains for 5. This implies that x^5 can be computed from x using only 3 multiplications:

$$\begin{aligned} x^2 &= x \cdot x, \\ x^4 &= x^2 \cdot x^2, \\ x^5 &= x^4 \cdot x. \end{aligned}$$

Another example: one of the shortest addition chains for 13 is $(1, 2, 4, 8, 12, 13)$.

For a detailed resource on the subject, see [Knu98, Section 4.6.3]. In this paper, we will only need the following property, in which we examine the largest values of n we can get for a fixed value of $\ell(n)$.

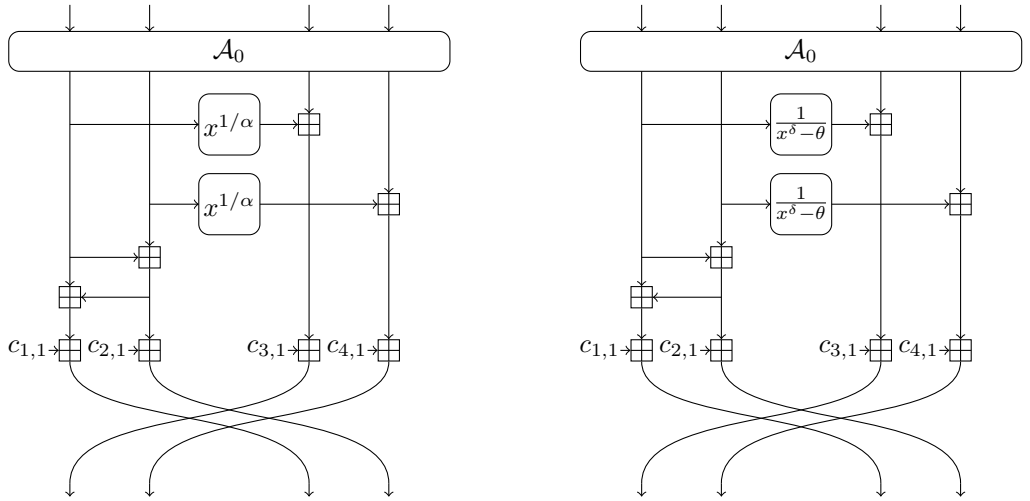
Proposition 11 (Largest n for fixed $\ell(n)$)**.** *Let $m \geq 2$. Then we have that*

1. *The largest value of n such that $\ell(n) = m$ is*

$$n = 2^m.$$

2. *The largest value of n such that n is odd and $\ell(n) = m$ is*

$$n = 2^{m-1} + 1.$$



(a) Lachesis. (b) Atropos.

Figure 6: First three layers of a Moirai with half-MDS layers.

Proof. The first part of the proposition can be deduced from the fact that $\ell(2^m) = m$ and $\ell(n) \geq \lceil \log_2(n) \rceil$ (more generally, in an addition chain, $k_i \leq 2^i$). This can be found in [Knu98, Section 4.6.3].

For the second part, we use induction. For $m = 2$, the only addition chains are $(1, 2, 3)$ and $(1, 2, 4)$, so the largest odd value for k_2 is 3. Suppose the proposition is true for all values less than a given $m \geq 3$. Then, consider an addition chain $(k_0, \ldots, k_i, \ldots, k_m)$ where k_m is odd and k_i is the largest odd number of the chain aside from k_m (which is always defined, as $k_0 = 1$). First, let us consider the case where $i \geq 2$. By induction, we have that $k_i \leq 2^{i-1} + 1$. Also, note that k_m must be a sum of an odd number and an even number, since it is odd. Therefore, k_m is upper bounded by the sum of k_i and the largest even number of the chain. Then, we consider two cases:

- If $i = m - 1$, both k_{m-1} and k_m are odd, and the largest even number of the chain is upper bounded by 2^{m-2}, by the first part of the proposition. Therefore,

$$k_m \leq 2^{m-2} + 1 + 2^{m-2} = 2^{m-1} + 1.$$

- If $i < m - 1$, the largest even number of the chain is k_{m-1}. We always have the upper bound $k_{m-1} \leq 2^{m-1-i} k_i$, and so

$$\begin{aligned} k_m &\leq k_i + k_{m-1} \\ &\leq 2^{i-1} + 1 + 2^{m-2} + 2^{m-1-i} \\ &< \sum_{a=0}^{m-2} 2^a \quad (\text{since } 2 \leq i < m) \\ &< 2^{m-1} + 1. \end{aligned}$$

This concludes the case where $i \geq 2$. We cannot have $i = 1$, as $k_1 = 2$. So, finally, let us consider the case where $i = 0$, i.e. all integers of the chain aside from k_0 are even. Again, k_m must be a sum between an odd integer and an even integer, but the only odd is 1. Therefore, we have that

$$k_m \leq k_{m-1} + 1 \leq 2^{m-1} + 1.$$

This concludes the induction. In fact, this bound is tight, and the addition chain $(1, 2, \ldots, 2^{m-1}, 2^{m-1}+1)$ yields the highest odd integer in an addition chain of size m. □

C Additional proofs of Section 5

Proof of Proposition 5 for MDS layers. We will construct a polynomial model that solves the CICO-b problem such that it is already a Gröbner basis in grevlex order, following the Freelunch [BBL+24] method. Denote $\mathbf{M}$ as the matrix that represents the linear part of the MDS layers. Ensuring that the system has the correct shape is quite tricky: the inputs and outputs of the Sboxes need to be specific affine functions of the variables. We can do the following: first, we represent the input of the CICO-b problem as a sum of vectors

$$x_{1,1}\mathbf{v}_{1,0} + \cdots + x_{b,1}\mathbf{v}_{b,0},$$

where $\mathbf{v}_{i,0} = (v_{i,0,1}, \ldots, v_{i,0,b}, 0, \ldots, 0) \in \mathbb{F}_p^{2b}$ is defined by a linear system of the form

$$\mathbf{M} \cdot \mathbf{v_{i,0}} = (\underbrace{0, \ldots, 0, 1}_{i}, \underbrace{0 \ldots, 0}_{b-i}, \underbrace{*, \ldots, *}_{b})$$

where 1 is at the i-th coordinate[11], i.e. $\mathbf{M} \cdot \mathbf{v_{i,0}}$ is equal to 0 on left-side coordinates except at i where it is equal to 1.

Now, we proceed by induction. Let $1 \leq j \leq n_r$ and $1 \leq i \leq b$. Suppose that at the input of the j-th $\mathcal{S}$ layer, the i-th branch is a degree-1 polynomial, denoted $L_{i,j}$, in $\{x_{i,j-1}\} \cup \{x_{i',j'} : 1 \leq i' \leq b, 0 \leq j' < j-1\}$ (this is true for $j = 1$ due to the above paragraph). We now introduce variables $x_{i,j}$ for $1 \leq i \leq b$ so that the state after $\mathcal{S}$ will be of the form

$$x_{1,j}\mathbf{v}_{1,j} + \cdots + x_{b,j}\mathbf{v}_{b,j} + \mathbf{w},$$

where

- $\mathbf{v}_{i,j} = (0, \ldots, 0, v_{i,j,1}, \ldots, v_{i,j,b}) \in \mathbb{F}_p^{2b}$ is defined by the same kind of linear system as earlier
$$\mathbf{M} \cdot \mathbf{v}_{i,j} = (\underbrace{0, \ldots, 0, 1}_{i}, \underbrace{0 \ldots, 0}_{b-i}, \underbrace{*, \ldots, *}_{b}),$$
so as to satisfy the conditions for the next induction step.
- $\mathbf{w}$ is the concatenation of two vectors of size b, $(*, \ldots, *) \parallel (0, \ldots, 0)$, where each $*$ represents a degree-1 polynomial in the variables of the set $\{x_{i',j'} : 1 \leq i' \leq b, 1 \leq j' \leq j-1\}$: these are the values of the branches that are not modified by the $\mathcal{S}$ layer.

For each $x_{i,j}$ variable, there will thus be a corresponding equation

$$(x_{i,j-1} + L_{i,j}(x_{i',j'} : 1 \leq i' \leq b, 0 \leq j' < j-1))^\beta = L'_{i,j}(x_{i',j'} : 1 \leq i' \leq b, 0 \leq j' \leq j),$$

where $L_{i,j}$ and $L'_{i,j}$ are polynomials of degree 1. The leading monomial of this equation is $x_{i,j-1}^\beta$, assuming that we use the grevlex order with a lexicographical order such that, if $j_1 < j_2$, then $x_{i_1,j_1} < x_{i_2,j_2}$ for all i_1, i_2, j_1, j_2. By induction, we've fully defined all variables. The last b equations are affine and can be reduced so that their leading monomials are precisely $\{x_{i,n_r} : 1 \leq i \leq b\}$. In doing so, we've ensured that the set of leading monomials of the model with the appropriate grevlex order is exactly $\{x_{i,j}^\beta : 1 \leq i \leq b, 0 \leq j < n_r\} \cup \{x_{i,n_r} : 1 \leq i \leq b\}$. They are coprime leading monomials, therefore they define a Gröbner basis, and the ideal degree is the product of the degrees of the leading monomials (see [BBL+24, Prop.4]), and we can finally deduce Equation (3). □

[11] Here, the index of the first coordinate is 1.

D Griffin

Here we give some results on Griffin [GHR+23]. We did not include it in the body of the paper because there is a lack of analysis (or even conjectures) of D_I outside of the CICO-1 case, which limits how far we can push its ALFOMs analysis. Griffin is a family of permutations that combines a modified Feistel scheme, called "Horst", with a "Rescue-like" SPN. The current cryptanalytic works that tackle Griffin [BBL+24, BBB+25] provide a sound analysis of the polynomial systems, but only for the CICO-1 problem. Therefore, we will only consider the case $k = 1$. Note that the Griffin authors only listed instances where $p \approx 2^{256}$ and $\lambda = 128$ (see [GHR+23, Table 2]), so it is not too constraining to only study CICO-1.

Taking the results from the Freelunch paper, we have that

$$D_I(\alpha, b, r) = \begin{cases} (\alpha(2\alpha+1))^{r-1} & \text{for } b \in \{3, 4\}, \\ 3(\alpha(2\alpha+1))^{r-2} & \text{for } b = 8, \\ (6\alpha+3)(\alpha(2\alpha+1))^{r-3} & \text{for } b \geq 12. \end{cases}$$

For the number of constraints, we again refer to [BBC+23] – the original Griffin authors computed, for the number of R1CS constraints of a power-α map, $\lfloor \log_2(\alpha) \rfloor + hw(\alpha) - 1$, a tight upper bound for $\ell(\alpha)$.

$$\begin{aligned} C_{R1CS}(\alpha, b, r) &= (\ell(\alpha) + b - 2) \cdot 2r, \\ C_{AIR}(\alpha, b, r) &= \alpha \cdot b \cdot r. \end{aligned}$$

Now, we can deduce the ALFOMs for Griffin:

$$\eta_{R1CS}(\alpha, b, r) = \begin{cases} \dfrac{(r-1)\log_2(\alpha(2\alpha+1))}{2r(\ell(\alpha)+b-2)} & \text{for } b \in \{3, 4\}, \\ \dfrac{\log_2(3) + (r-2)\log_2(\alpha(2\alpha+1))}{2r(\ell(\alpha)+6)} & \text{for } b = 8, \\ \dfrac{\log_2(6\alpha+3) + (r-3)\log_2(\alpha(2\alpha+1))}{2r(\ell(\alpha)+10)} & \text{for } b \geq 12. \end{cases}$$

Here are some observations:

- Much like Poseidon, the ALFOMs of Griffin decreases with the number of branches. This becomes obvious when one considers that D_I does not increase with b, but the number of constraints does. It is as of yet unclear whether the ALFOMs would increase with k. However, Poseidon mitigates this issue with its many partial layers. For Griffin, it is clear that at constant $p \approx 2^{256}$ and security parameter $\lambda = 128$ (implicitly assuming that the security relies on the hardness of CICO-1), the best ALFOMs is achieved at the lowest number of branches, $b = 3$. This is doubly true when taking into account the number of skipped rounds, which increases with b.
- Asymptotically, as the number of rounds increases, we have the following:

$$\eta_{R1CS}(\alpha, b, r) \underset{r\to\infty}{\sim} \begin{cases} r\dfrac{\log_2(\alpha(2\alpha+1))}{2(\ell(\alpha)+b-2)} & \text{for } b \in \{3, 4\}, \\ r\dfrac{\log_2(\alpha(2\alpha+1))}{2(\ell(\alpha)+6)} & \text{for } b = 8, \\ r\dfrac{\log_2(\alpha(2\alpha+1))}{2(\ell(\alpha)+10)} & \text{for } b \geq 12. \end{cases}$$

  However, keep in mind that the round skips here are particularly powerful (3 rounds out of 9 skipped for some instances where $b \geq 12$), so the asymptotic analysis will yield a looser approximation, compared to the other primitives in this paper.

E Experimental data on Atropos

Table 4: Experimental data of Atropos's ideal degree for the model we describe in the proof of Prop. 9. A (✓) next to D_I means that the Bézout bound is reached, a (✗) means it is not reached.

(b, n_r, δ)	Bézout bound	D_I (MDS layers)	D_I (half-MDS internal layers)
(1, 1, 2)	3	3 (✓)	3 (✓)
(1, 1, 3)	4	4 (✓)	4 (✓)
(1, 2, 2)	9	9 (✓)	9 (✓)
(1, 2, 3)	16	16 (✓)	16 (✓)
(1, 3, 2)	27	27 (✓)	23 (✗)
(1, 3, 3)	64	64 (✓)	55 (✗)
(1, 4, 2)	81	81 (✓)	57 (✗)
(1, 4, 3)	256	256 (✓)	184 (✗)
(1, 5, 2)	243	243 (✓)	139 (✗)
(1, 6, 2)	729	729 (✓)	337 (✗)
(2, 1, 2)	9	9 (✓)	9 (✓)
(2, 1, 3)	16	16 (✓)	16 (✓)
(2, 2, 2)	81	81 (✓)	81 (✓)
(2, 2, 3)	256	256 (✓)	256 (✓)
(2, 3, 2)	729	729 (✓)	537 (✗)
(3, 1, 2)	27	27 (✓)	27 (✓)
(3, 1, 3)	64	64 (✓)	64 (✓)
(3, 2, 2)	729	729 (✓)	729 (✓)
(4, 1, 2)	81	81 (✓)	81 (✓)
(4, 1, 3)	256	256 (✓)	256 (✓)
(5, 1, 2)	243	243 (✓)	243 (✓)
(6, 1, 2)	729	729 (✓)	729 (✓)