The Poseidon Hash Function formalisation in Lean 4

Ashvni Narayanan and Daniel Rogozin, for Yatima Inc

August 31, 2022*

1 Introduction

In this document, we describe the Poseidon formalisation in Lean 4. Initially, Poseidon has been introduced as a family of hash functions based on the permutations $Poseidon^{\pi}$, see [1]. This approach utilises the Hades strategy [2] according to which (TODO: explain the Hades strategy with a couple of sentences).

2 Formalisation itself

We need some preliminaries first. Let p, t be natural numbers and p is prime. As usual, $\mathbb{Z}_p$ stands for prime field of order p. In the original text, it is also assumed that $\lceil \log_2(p) \rceil = n$ for some $n > 0$.

We introduce these requirements in Lean 4 by declaring the following variables.

```
variable (p t : ℕ) [Fact p.Prime] [Field (Zmod p)] [Fintype (Fin_x t)]

def ARC (c a : Fin_x t → Zmod p) (i : Fin_x t) : Zmod p := (a i) + (c i)

def R_f_round (S_box' : Zmod p → Zmod p) (c : Fin_x t → Zmod p)
  (MDS' : Matrix (Fin_x t) (Fin_x t) (Zmod p)) (a : Fin_x t → Zmod p) : Fin_x t → Zmod p :=
  Matrix.mulVec_x MDS' (λ i => S_box' (ARC p t c a i))

    def R_p_round (S_box' : Zmod p → Zmod p) (c : Fin_x t → Zmod p)
    (MDS' : Matrix (Fin_x t) (Fin_x t) (Zmod p)) (a : Fin_x t → Zmod p) : Fin_x t → Zmod p :=
    Matrix.mulVec_x MDS'
      (λ i => dite ((i : ℕ) = 0) (λ _ => S_box' (ARC p t c a i)) (λ _ => ARC p t c a i))

def P_perm (R_f R_p : ℕ) (S_box' : Zmod p → Zmod p) (c a : Fin_x t → Zmod p)
  (MDS' : Matrix (Fin_x t) (Fin_x t) (Zmod p)) : Fin_x t → Zmod p :=
  (R_f_round p t S_box' c MDS')^[R_f] ((R_p_round p t S_box' c MDS')^[R_p]
  ((R_f_round p t S_box' c MDS')^[R_f] a))

def add_to_state (r cap : ℕ) (m : Fin_x r → Zmod p)
  (a : Fin_x t → Zmod p) (h : t = r + cap) : Fin_x t → Zmod p :=
  λ i => dite ((i : ℕ) < r) (λ h => a i + m (Fin_x.castLt i h)) (λ h => a i)

def P_hash (R_f R_p r o cap : ℕ) (hr : 1 ≤ r) (S_box' : Zmod p → Zmod p)
  (c : Fin_x (r + cap) → Zmod p)
  (MDS' : Matrix (Fin_x (r + cap)) (Fin_x (r + cap)) (Zmod p)) (ho : o ≤ r + cap)
  (k : ℕ) (a : Fin_x (k * r + (r + cap)) → Zmod p) : Fin_x o → Zmod p
```

*The Lean 3 formalisation is by Ashvni Narayanan. Daniel Rogozin prepared the Lean 4 version and its text description.

References

[1] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. Poseidon: A new hash function for {Zero-Knowledge} proof systems. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 519–535, 2021.

[2] Lorenzo Grassi, Reinhard Lüftenegger, Christian Rechberger, Dragos Rotaru, and Markus Schofnegger. On a generalization of substitution-permutation networks: The hades design strategy. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 674–704. Springer, 2020.