The sponge construction [BDPV07] provides a generic framework for building variable-length hash functions from a fixed-width permutation P. It is originally defined to operate on bits, i.e., using a b-bit permutation P and operating on a state of size b bits that is split into an inner part of c bits and an outer part of r bits, processing messages by absorbing data into the outer part r bits at a time, and then squeezing digests r bits at a time. The message is first *injectively* padded into r-bit blocks:

$$M \mapsto \mathtt{pad}(M) = M\|1\|0^* \,. \tag{1}$$

I.e., we add a single 1 and a sufficient number of 0's so that the padded message is of length a multiple of r. We call this type of padding "10-padding". The overall function is depicted in Figure 1.

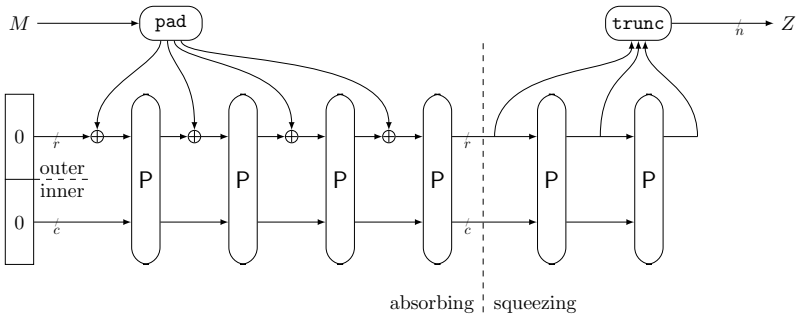

Figure 1: Sponge construction, with a requested output of n bits.

The leading security model for hashing is indifferentiability from a random oracle [MRH04, CDMP05]. This particularly implies that the hash function behaves like a random oracle up to the proven bound and that it can be used as such. Bertoni et al. [BDPA08] proved indifferentiability of the sponge, up to $\sqrt{2^{c/2}}$ queries to P when this underlying permutation P is modeled as a random permutation. Andreeva et al. [AMP10, Appendix A] demonstrated that classical security follows from indifferentiability. Simply said, the sponge construction achieves collision resistance, preimage resistance, and more.

These results also hold for the overwrite sponge, where the message blocks $M_1, M_2, \ldots$ are overwriting the outer part instead of being added into it. The results also simply generalize to the case where the inner and outer parts are defined over larger fields.

What About 0-Padding? Suppose we instead pad

$$M \mapsto \mathtt{pad}(M) = M\|0^* \,. \tag{2}$$

We call this type of padding "0-padding". In this case, picking any M of size $r-1$ bits and setting $M' = M\|0$ yields $\mathtt{sponge}(M) = \mathtt{sponge}(M')$. This is a collision, and hence a break on the security of the hash function. The same observation applies if the message blocks are being overwritten into the state, with the difference that one may have to take $M' = M\|C$ where C is the actual state bit at position r.

Clever Padding. Clearly, 10-padding of (1) incurs an extra permutation call in case the original M happens to be a multiple of r. To avoid this, one can do the following:

- If M *is not* of size a multiple of r: perform 10-padding of (1);
- If M *is* of size a multiple of r: perform simple 0-padding of (2) and transform the inner part by adding 1 to any element.

Call this function $\mathtt{sponge}'$. This is a direct simplification of the `sponge-pi` construction of Lefevre et al. [LBM25, Section 3.1], and it is indifferentiable from (i.e., behaves like) a random oracle. In fact, their construction is more general but right after their "Constraint 2" they give exactly this example. Collision resistance, preimage resistance, ..., follow, and the results also apply to the overwrite sponge, where the padded message blocks overwrite the outer part (but addition of 1 to the inner part remains addition and not overwrite).

What About Transforming Inner Part for Partial Messages Instead? If we would do the "transform the inner part by adding 1 to any element" for partial messages instead of full messages, the construction would actually be *generically insecure*. The reason is that the sponge supports squeezing multiple blocks, and we can observe that for any message M of size a multiple of r:

$$\text{"digest blocks } 2,3,4,\ldots \text{ of } \mathtt{sponge}'(M)\text{"} = \text{"digest blocks } 1,2,3,\ldots \text{ of } \mathtt{sponge}'(M\|0^r)\text{"} \,.$$

This is something you wouldn't expect of a random oracle. (The function may still be collision resistant but we have no proof for this.)

References

[AMP10] Elena Andreeva, Bart Mennink, and Bart Preneel. Security Reductions of the Second Round SHA-3 Candidates. In Mike Burmester, Gene Tsudik, Spyros S. Magliveras, and Ivana Ilic, editors, *Information Security - 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers*, volume 6531 of *Lecture Notes in Computer Science*, pages 39–53. Springer, 2010.

[BDPA08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the Indifferentiability of the Sponge Construction. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages 181–197. Springer, 2008.

[BDPV07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge Functions. Ecrypt Hash Workshop 2007, May 2007.

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård Revisited: How to Construct a Hash Function. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.

[LBM25] Charlotte Lefevre, Mario Marhuenda Beltrán, and Bart Mennink. To Pad or Not to Pad? Padding-Free Arithmetization-Oriented Sponges. *IACR Trans. Symmetric Cryptol.*, 2025(1):97–137, 2025.

[MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.