PQC: R-Propping a Chaotic Cellular Automata

Pedro Hecht

Information Security Master, School of Economic Sciences,
School of Exact and Natural Sciences and Engineering School (ENAP-FCE),
University of Buenos Aires, Av. Cordoba 2122 2nd Floor,
CABA C1120AAP, República Argentina
phecht@dc.uba.ar, qubit101@gmail.com

Abstract. Post-quantum cryptography (PQC) has a well-deserved NIST status. Our approach (R-Propping), replaces all numeric field arithmetic with GF(2^8) field operations. This method yields both classical and quantum secure protocols. The present work is dedicated to strengthening a chaotic Wolfram Class III cellular automata and discuss its usability as a cryptographical secure PRBG (pseudorandom bit generator), a building block for stream-ciphers, hashing, and other random numbers requiring protocols.

Keywords: Post-quantum cryptography, algebraic rings, combinatorial group theory, R-propping.

1 Introduction

1.1 Brief background of Post-Quantum Cryptography

Since the first generation of public-key cryptography (PKC) was introduced by Diffie and Hellman, many schemes have been proposed and broken. Recently Post-Quantum Cryptography (PQC) attempts to develop cryptographic protocols that are simultaneously resistant to classical and to quantum computing attacks using Shor's polynomial-time algorithm or Grover's algorithm. Today NIST conducts a 3rd round survey to solicit, evaluate, and standardize one or more quantum-resistant public-key cryptographic algorithms [1].

1.2 NIST PQC Proposals based on Combinatorial Group Theory

Currently, no NIST proposal is based on OWF (one-way functions) extracted from the CGT (Combinatorial Group Theory) [2]. Only one 1st round proposal of this kind and developed by us was presented and detached because of the selected faulty platform. Today the R-Propped version [3] corrects this problem and waits until the following PQC survey. The R here refers to Rijndael. The main advantages of this class of protocols are cryptographic security, small-sized keys, fast operation times, and easy hardware portability. Other works of the author covering this subject can be found at [4 to 12].

1.3 The motivation of the present work

There was an attempt to apply field operations to CA searching, although using reversible AC that exhibits group properties [13]. In our case, we focus on chaotic AC as an attempt to obtain pseudorandom generators. Chaotic 1D cellular automata (CA) (i.e Wolfram's Class III Rule 30) [14 to 18] has been proposed a long time ago as PRBG (pseudo-random bit generators) [19]. No formal demonstration of their cryptographic security has been developed but many theoretical arguments have been provided to support this conclusion [18]. Our goal is to provide an empirical solution to this question: (a) extend their use from binary cells to GF(2^8) byte represented internal states and replace *mod 2* operations (see eq. 2) with field sum and field product, an R-Propping procedure of the state-updating formula and (b) study the statistical, informational and complexity properties of the transformed protocol. The ultimate target is to develop stream-ciphers, asymmetric ciphers, digital signatures, hashing, and other PRBG requiring protocols.

2 Preliminaries

2.1 Class III Rule 30 CA original structure

The 1-Dimensional Boolean Rule 30 [14, 17] evolution has the following functional representation where *i*-subscript represents the position at any site and the apostrophe next-generation value.

$$a'_i = a_i . xor . (a_i . or . a_{i+1}) \qquad (1)$$

or equivalently

$$a'_i = [a_{i-1} + a_i + a_{i+1} + a_i a_{i+1}] \, (mod\ 2) \qquad (2)$$

The graphic representation and initial evolution are depicted in Table 1. The random conduct of the central state is as Table2. displays.

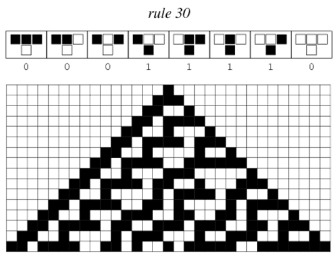
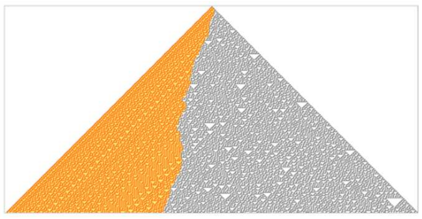
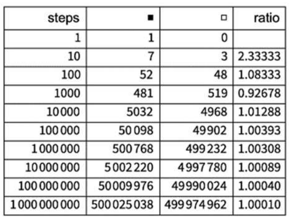


Table 1. The rule defines the central successor state for each contiguous triplet site. Here starting from a canonical {0....,1,0.....} state. black cells are 1 and white cells 0 internal states. The decimal format of the next-step site value whose binary is {00011110} is the naming rule number.





steps	■	□	ratio
1	1	0	
10	7	3	2.33333
100	52	48	1.08333
1000	481	519	0.92678
10000	5032	4968	1.01288
100000	50098	49902	1.00393
1000000	500768	499232	1.00308
10000000	5002220	4997780	1.00089
100000000	50009976	49990024	1.00040
1000000000	500025038	499974962	1.00010

Table 2. Periodic (yellow) and aperiodic (grey) behavior of Boolean Rule 30 CA with no boundary limits starting from the canonical {0....,1,0.....} state. If a random valued initial state is used, then only the pseudorandom behavior is obtained. At right successive tallying one and zero values of the central column. The {1,0} succession is represented as the (OEIS A051023) sequence. If the ratio-limit is exactly one remains an open question [16].

The chaotic property of Rule 30 was intensively studied by Wolfram [14,16].

2.2 R-Propped Rule 30 CA

Adapting Eq. 2 to the new format, we use:

a. The dimension of the CA is finite and fixed.
b. To avoid border effects, a circular ring or torus-like structure was adopted.
c. Each site contains a $\mathbb{Z}_{256}$ value.
d. Adapted R-Propped function using field operations at Eq. 3.

$$a'_i = [a_{i-1} \oplus a_i \oplus a_{i+1} \oplus (\, a_i \odot a_{i+1})] \;\Leftarrow\; GF[2^8, \oplus, \odot] \qquad (3)$$

or as a Mathematica 12 with new defined functions at Eq. 4

$$\text{Rule30}[\text{x}\_, y\_, z\_] := \text{Sum}[x, y, z, Mult[y, z]] \qquad (4)$$

If we restrict values to the prime Subfield GF(2), the CA duplicated the original output.

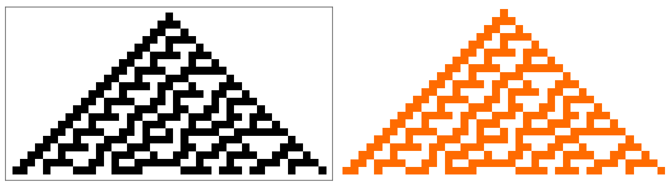


Table 3. At left the first 20-generations of the original Rule 30 CA and at right the equivalent of the R-propped version restricted to the GF(2) subfield, both starting at canonical {0...,1,0...} state.

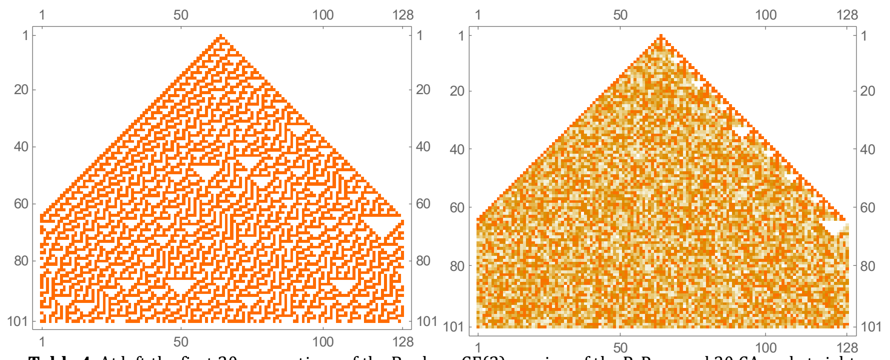


Table 4. At left the first 20-generations of the Boolean GF(2) version of the R-Propped 30 CA and at right the equivalent evolution of the full GF(2^8) version, both starting at canonical {0...,1,0...} state. The random behavior is enhanced at the right.

3 Experimentally observed properties of R-Propped Rule 30 CA

3.1 Statistic of site values

Running the R-Propped Rule 30 CA and recording the frequency of bytes at the center site of a 128-byte ring, Table 5. shows the obtained results. The same data are displayed in Table 6. Observe the normal distribution of overall frequencies and uniform distribution for the byte values.

This behavior is consistent with a regular random generator (PRBG) and further empirical evidence is provided in the following sections.

```
4162, 4064, 4110, 4091, 4137, 4068, 4205, 4039, 4023, 4092, 4016, 4097, 4146, 4046, 4120,
4200, 4080, 4219, 4259, 4072, 4118, 4196, 4106, 4046, 4141, 4189, 4060, 4133, 4096, 4163,
4082, 4146, 4004, 4095, 4056, 4083, 4054, 4193, 4025, 4118, 4065, 4111, 4005, 4115, 4075,
4218, 4115, 4098, 4203, 4042, 4052, 4106, 4048, 4104, 4131, 4024, 3998, 4070, 4041, 4064,
4015, 3953, 4167, 4152, 4067, 4130, 4082, 4026, 4030, 4074, 4046, 4140, 4105, 4081, 4102,
4004, 4086, 4055, 4079, 4142, 4094, 4102, 4197, 4020, 3954, 3982, 4156, 4054, 3972, 4171,
4168, 4179, 4198, 3940, 4114, 3974, 4050, 4178, 4127, 4113, 4137, 4170, 4098, 4177, 4078,
3995, 4133, 4099, 4156, 4068, 4152, 4007, 4066, 4113, 4201, 4200, 4073, 4035, 4030, 4102,
4155, 4040, 4097, 4043, 4166, 4174, 4037, 4096, 4059, 4073, 4265, 4134, 4255, 4122, 4058,
4109, 4110, 4116, 4179, 4094, 4189, 4000, 3961, 4147, 4039, 4161, 4146, 4098, 3978, 4096,
4142, 4101, 4194, 4189, 3976, 4083, 4065, 4098, 4106, 4026, 4035, 4113, 4082, 4098, 4013,
4132, 4170, 4106, 4137, 4127, 4127, 4191, 4081, 4093, 4067, 4095, 4124, 4118, 4092, 4046,
4136, 4120, 4097, 4071, 4150, 4087, 4092, 4176, 4060, 4189, 4070, 4182, 4140, 4172, 4099,
4085, 3987, 4196, 4052, 4008, 4177, 4239, 4117, 4070, 4021, 4102, 4075, 4095, 4052, 4159,
4136, 3969, 4061, 4105, 4101, 4183, 4131, 4079, 4043, 4133, 3941, 4029, 3988, 4012, 4089,
4042, 4146, 4030, 4136, 4200, 4141, 4029, 3967, 3926, 4161, 4064, 3991, 4098, 4156, 4089,
4019, 4112, 3978, 4216, 4049, 4177, 4035, 3968, 3930, 4197, 4080, 4125, 4184, 4158, 4198,
4071
```

Table 5. Frequency of bytes [0-255] registered at the center site (64) of a random seeded R-Propped 30 CA of dimension=128 bytes during 2^{20} generations.

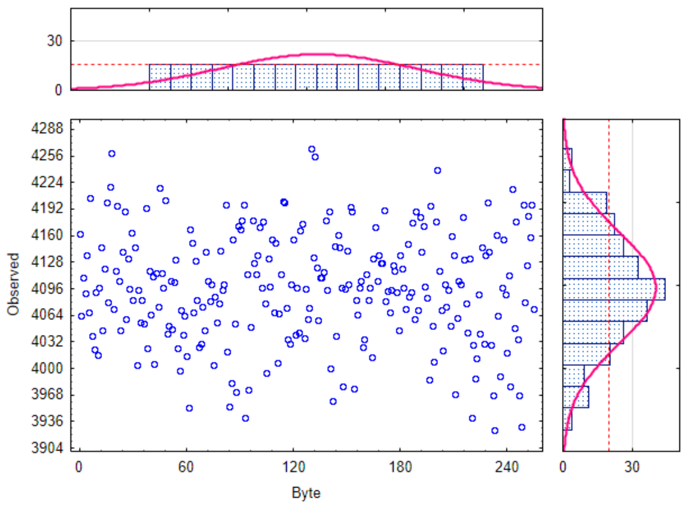


Table 6. Graphic representation of frequencies of bytes [0-255] observed at the center site (64) of a random seeded R-Propped 30 CA of dimension=128 during 2^{20} generations. At right the histogram of accumulated frequencies adjusted to a normal distribution (full red) and a uniform distribution (dashed red), and at the upper margin the corresponding histogram and adjustments of the [0-255] byte range.

3.2 Bit profiling of site values

To display the internal bit distribution of the CA output, it was applied a profiler run [20] of the same list used in the previous section. Results are presented in Table 7. and they confirm the regular random byte production.

Bit position	Total of Bit=1	%
0	519501	12.4
1	526015	12.6
2	516267	12.4
3	527517	12.6
4	521049	12.5
5	526084	12.6
6	518858	12.4
7	518026	12.4

Table 7. Bit profiler results of the byte list processed in the previous section. The total number of 1-bit was 4173317 (49.7%) and the total number of 0-bit was 4215299 (50.3%), so the 1/0 ratio was 0.990041 (Chi-square=12232.40, entropy=7.991567 bits/byte, sample size=1048577 bytes).

3.3 Site entropy of successive generations

Here we apply Shannon entropy [21] to an R-Propped Rule 30 CA 1024-binary sites (128-bytes) with a random seed. The formula obtains the mean entropy per binary site, minimum value=0 bits and maximum=1 bit. When the uncertainty becomes the maximum, a perfect uniform random probability density distribution (PDF) is attained, which reflects a true chaotic behavior if we add the impossibility of any deterministic prediction.

As table 8. shows, the entropy of the GF(2) variant increases along with successive generations and approaches to the theoretical limit of 1 bit/site, a perfectly random distribution.

It could be verified that the AC dimension (number of sites in the ring) has a fundamental role in defining the entropy measures. As Table 9. shows for GF(2^8), increasing the dimension approach the entropy to the theoretical limit of 8 bit/site and at the same time reduces the variability of the limiting profiles. These are strong reasons to support the hypothesis of the random output of the R-Propped Rule 30 AC.

Table 10. shows how the site entropy at the last generation of a 100-generations run, increases with the dimension in bytes of the R-Propped Rule 30 CA and at the same time infer that a 4096-bytes (2^{14} sites) automata suffice to give a good random performance for any real-life application.

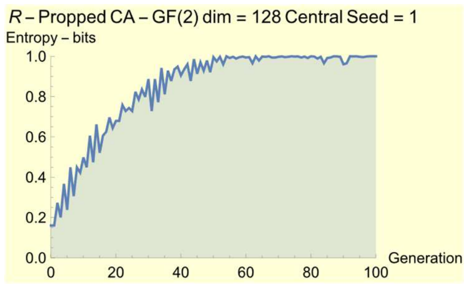
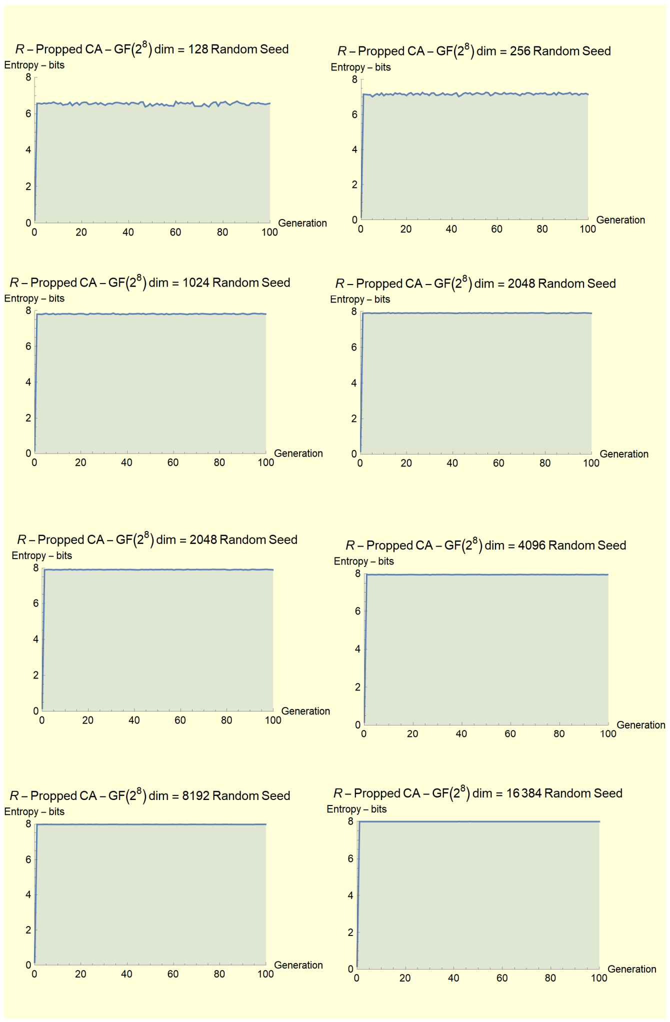
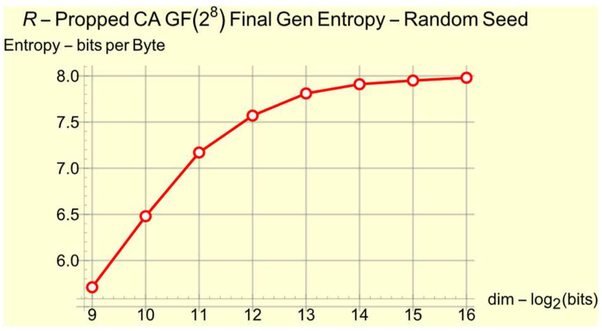


Table 8. For 1024bit sites (128 bytes) GF(2) Rule 30, the evolution from a central 1 seed, entropy increases until it reaches the theoretical limit of 1bit/site. The graph shows generation mean entropy per site from starting configuration 100 generations ahead.



. **Table 9**. For increasing dimension (bytes) of GF(2^8) Rule 30, the evolution from random seeds, entropy increases until it reaches the theoretical limit of 8 bit/site. The graph shows the generation means entropy per site from starting configuration to 100 generations ahead.



. **Table 10**. Dependence of the final generation site entropy for increasing dimension (bytes) of GF(2^8) Rule 30. In each case, 100-generations are studied. This result allows the selection of a practical dimension of 4096-bytes to obtain a reasonable high entropy for real-life applications.

3.4 Site entropy variability of successive generations

Table 11. displays the mean entropy variability along 100-generations shows that it depends on the ring dimension. This behavior could also be inferred from Table 6. Fluctuations.

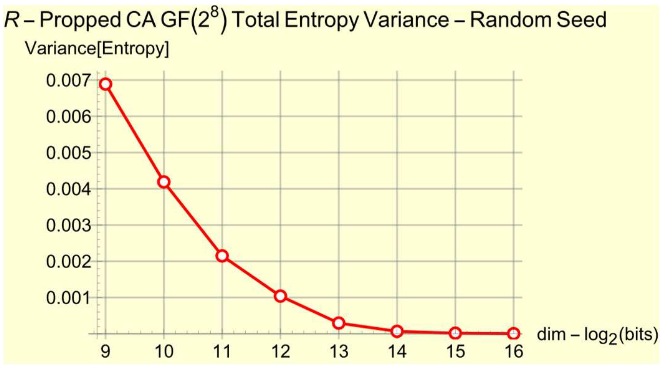


Table 11. Dependence of the total site entropy variance for increasing dimension (bytes) of GF(2^8) Rule 30. This result allows the selection of a practical dimension of 4096-bytes to obtain a reasonable high entropy for real-life applications.

3.5 Computation time for the 100-generation runs

Table 12. shows the mean run time for 100 successive generations as a function of the ring dimension. All programs are run as Mathematica 12 interpreted source code running as-is on an Intel®Core™i5-5200U CPU 2.20 GHz.

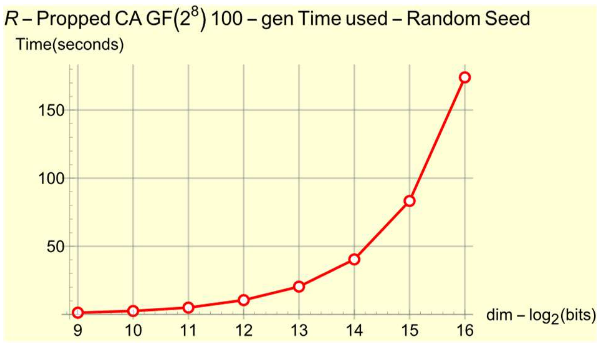


Table 12. The computation time of 100-generation runs of a Mathematica interpreted code. The optimal dimension for concrete applications should be chosen as high as the restrictions allow.

3.6 Computational complexity

Time depending computational complexity of algorithms is usually calculated constructively, summing up the elementary step operations involved [22]. When the code is complex or unknown, an alternative empirical approach consists of comparing computation time (in arbitrary units) against the number of input-bits for increasing input sizes. In our case, we choose this method.

In Table 13. we show the resulting linear adjustment for the R-Propped CA output, not only the pattern results polynomial in time but it behaves linear, so the empirical time complexity is $\mathbb{O}(n)$. This value is compliant with the value informed by Wolfram for the Boolean CA Rule 30, which suggests an order of $\mathbb{O}(n^p)\ ; p < 2$ [16].

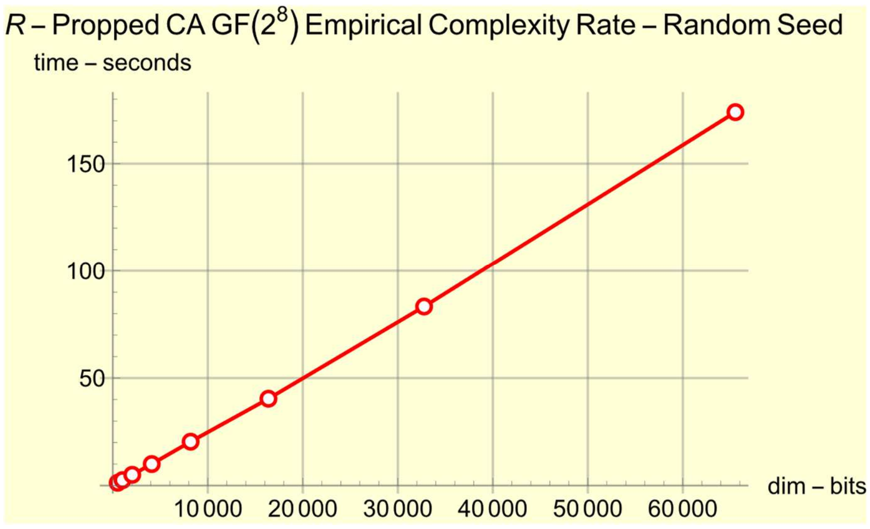


Table 13. The computation time of 100-generation runs of a Mathematica interpreted code as a function of bits generated by the R-Propped Rule 30 CA. As the linear adjustment suggests, the operational complexity of the generating algorithm is $\mathbb{O}(n)$ or at least $\mathbb{O}(n^p)\ ; p < 2$ as [16] suggests.

3.7 Spectral Randomness Test of the R-Propped CA output

This particular test was selected following Donald Knuth's opinion, in his own words as follows [23]: *"...not only do all good [pseudorandom] generators pass this test, all generators are now known to be bad fail it. This is by far the most powerful test known, and it deserves particular attention."* To perform it, we used a discrete cosine transform-based method to test the randomness of a sequence of random integers developed in Mathematica language [24].

The examined sample was 8519680 bits. With an ***H_0: Tested PDF is randomly uniform***, the p-value was 0.962 (accept ***H_0***)

3.8 Semantic security and non-malleability for the R-Propped CA output

A direct way to attack a PRBG is to be capable of predicting the next bit with probability > ½, knowing any arbitrary historic sequence of the same source. We conjecture that the present solution lacks this weakness, so any bitstream would be indistinguishable from any other of the same length. With this property, the challenger exhibits semantic security and wins the indistinguishable game of chosen-plaintext attack (CPA), non-adaptive chosen-ciphertext attacks (CCA1), and adaptive chosen-ciphertext attacks (CCA2) against any adversary with randomized probabilistic polynomial-time (PPT) resources [25].

4 Conclusions

We present at this paper an application of the previously described R-Propping technique. The ultimate goal is to generate a general-purpose PRBG with evidence supporting a satisfactory statistical performance. The choice of a Chaotic CA has many advantages: chaotic output for the GF(2) subfield, fast operation-time, scalable, full-parametric structure, and cryptographic security. At the current time, no classical [26, 27] or quantum attack [28, 29] is at hand, and probable no one will shortly appear. Also, semantic security and non-malleability block any probability advantage to effectively predict the next-bit state.

Besides the fundamental work of Wolfram, many valuable authors have contributed to the study of applications of chaotic CA into cryptography [30, 31, 32, 33, and more].

No source code was included here, but a full Mathematica 12 source code is available upon request.

References

1. NIST PQC Security Categories, https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions
2. A. Myasnikov, V. Shpilrain, A. Ushakov, Non-commutative Cryptography and Complexity of Group-theoretic Problems, Mathematical Surveys and Monographs, AMS Volume 177, 2011
3. P. Hecht, “R-Propping of HK17: Upgrade for a Detached Proposal of NIST PQC First Round Survey”, https://eprint.iacr.org/2020/1217, DOI: 10.13140/RG.2.2.31287.96163, 7pp, 2020
4. P. Hecht, “Post-Quantum Cryptography(PQC): Generalized ElGamal Cipher over GF(251^8)”, ArXiv Cryptography and Security (cs.CR) http://arxiv.org/abs/1702.03587 & Journal of Theoretical and Applied Informatics (TAAI), 28:4, pp 1-14 (2016), http://dx.doi.org/10.20904/284001, 6pp, 2017
5. P. Hecht, “Post-Quantum Cryptography: A Zero-Knowledge Authentication Protocol”, ArXiv Cryptography and Security (cs.CR) https://arxiv.org/abs/1703.08630, 3pp (2017)
6. P. Hecht, “Post-Quantum Cryptography: S381 Cyclic Subgroup of High Order”, ArXiv Cryptography and Security (cs.CR) http://arxiv.org/abs/1704.07238 (preprint) & International Journal of Advanced Engineering Research and Science (IJAERS) 4:6, pp 78-86 (2017), https://dx.doi.org/10.22161/ijaers.4.6.10, 2017
7. P. Hecht, "PQC: Triple Decomposition Problem Applied To GL(d, Fp) - A Secure Framework For Canonical Non-Commutative Cryptography", Hecht P., ArXiv Cryptography and Security (cs. CR) https://arxiv.org/abs/1810.08983, DOI: 10.13140/RG.2.2.23240.78082, 9pp, 2018
8. "PQC: Extended Triple Decomposition Problem Applied To GL(d, Fp) – An Evolved Framework For Canonical Non-Commutative Cryptography", Hecht P., ArXiv Cryptography and Security (cs. CR), https://arxiv.org/abs/1812.05454v1, 2pp (2018), DOI: 10.13140/RG.2.2.20242.09926
9. P. Hecht, “Algebraic Extension Ring Framework for Non-Commutative Asymmetric Cryptography“, ArXiv Cryptography and Security (cs. CR), https://arxiv.org/abs/2002.08343, DOI: 10.13140/RG.2.2.25826.56002, 4pp, 2020
10. P. Hecht, "PQC: R-Propping of Public-Key Cryptosystems Using Polynomials over Non-commutative Algebraic Extension Rings", https://eprint.iacr.org/2020/1102, 10pp DOI: 10.13140/RG.2.2.25826.56002, 4pp, 2020
11. P. Hecht, “PQC: R-Propping of Burmester-Desmedt Conference Key Distribution System”,https://eprint.iacr.org/2021/024, DOI:10.13140/RG.2.2.22638.43846, 2021
12. P. Hecht, “PQC: R-Propping of a New Group-Based Digital Signature”, https://eprint.iacr.org/2021/270, DOI: 10.13140/RG.2.2.35795.91683, 2021
13. D. Mukhopadhyay, D. Roy Chowdhury, Cellular Automata-based cryptosystem employing Galois Field algebra, The Japan Society of Mechanical Engineering, NII Electronic Library Service, No 01-63, 2001

14. S. Wolfram, “Statistical Mechanics of Cellular Automata”, Rev.Mod. Phys, 55, 601-644, 1983
15. S. Wolfram, A New Kind of Science, Champaign, IL: Wolfram Media, pp. 29-30, 52, 57, 317 and p. 871, 2002
16. S. Wolfram, https://writings.stephenwolfram.com/2019/10/announcing-the-rule-30-prizes/, 2019
17. S. Wolfram, Rule 30, https://mathworld.wolfram.com/Rule30.html
18. S. Wolfram, Random Sequence Generation by Cellular Automata, Advances In Applied Mathematics 7,123-169, 1986
19. A. Menezes, P. van Oorschot, and S.Vanstone, ”Handbook of Applied Cryptography”, CRC Press, 1997.
20. D. Stanislawek, proFILEr v1.01, Copyright (C)2004 (request a copy if needed)
21. T. M. Cover, J. A. Thomas, Elements of Information Theory, Wiley Interscience Pub., 2nd Ed, 2006
22. E. Bach, J. Shallit, Algorithmic Number Theory, Vol. 1, “Efficient Algorithms”, MIT Press, 2nd Ed, 1996
23. D. E. Knuth, The Art of Computer Programming, Vol 2, Seminumerical Algorithms, Addison-Wesley, 3rd Ed, 1997
24. E. Blumenthal, N. Blumenthal, Spectral Randomness Test, Wolfram Function Repository, https://resources.wolframcloud.com/FunctionRepository/, 2021
25. J. Katz, Y. Lindell, Introduction to Modern Cryptography, Chapman & Hall/CRC, 2008
26. V. Roman'kov, Cryptanalysis of a combinatorial public-key cryptosystem, De Gruyter, Groups Complex. Cryptology. 2017.
27. A. Ben Zvi, A. Kalka, B. Tsaban, Cryptanalysis via algebraic spans, CRYPTO 2018, Lecture Notes in Computer Science 10991 (2018), 255-274. https://doi.org/10.1007/978-3-319-96884-1_9
28. P. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”, SIAM J. Comput., no. 5, pp. 1484-1509, 1997.
29. L.K. Grover, “A fast quantum mechanical algorithm for database search”. In Proc. 28th Ann. ACM Symp. on Theory of Computing (ed. Miller, G. L.) 212–219, ACM, 1996.
30. S. Nandi, B.K. Kar, and P. Pal Chaudhuri. Theory and applications of cellular automata in cryptography. IEEE Transactions on Computers, 43(12):1346–1357, 1994
31. M. Tardivo Filho, M. A. Amaral Henriquez, Estudo sobre a Aplicação de Autômatos Celulares Caóticos em Criptografia, https://www.researchgate.net/profile/Mauro-Tardivo-Filho/publication/235675864
32. Tomassini, M.; Sipper, M.; Perrenoud, M., "On the generation of high-quality random numbers by two-dimensional cellular automata". IEEE Transactions on Computers. 49 (10): 1146–1151. doi:10.1109/12.888056., 2000
33. J. Kroc; F. Jiménez-Morales, J. L Guisado, M. C. Lemos, J. Tkáč,. "Building Efficient Computational Cellular Automata Models of Complex Systems: Background, Applications, Results, Software, and Pathologies". Advances in Complex Systems. 22 (5): 1950013 (38 pages). doi:10.1142/S0219525919500139., 2019