Cuckoo-Cycle Profiling

Karl-Johan Alm <karl@dglab.com>
DG Lab

May 1, 2017

Contents

1 Introduction

This document details a series of profiling tests done on Cuckoo-Cycle[1]. Parts of it are most likely not comprehensible without having read the white paper at least cursively. The tests were done in order to find optimal default parameters for a proof of work challenge with the aim of maximizing RAM usage while keeping solution time low enough that nodes could fine tune the difficulty without generating overly difficult challenges.

2 Limitations in Existing Hardware

This section lists existing hardware and their limitations in particular in terms of available RAM.

hardware		CPU		RAM (MB)	
device	model	cores	speed/core	total	available
Raspberry Pi	A+	1	700 MHz	256	-[2]
Raspberry Pi	Zero*	1	1 GHz	512	~100[2]
Raspberry Pi	2	4	900 MHz	1024	648[2]
Raspberry Pi	3 Model B	4	1.2 GHz	1024	648[2]
Desktop	2017	-	-	8192	*
Laptop	2017	-	-	4096	*
ASIC[3]	-	-	-	0	-

Table 1: Hardware comparison.

To facilitate most of the RAM-enabled devices, the proof of work must consume a maximum of approximately 100 MB of RAM. This excludes the Raspberry A+ as it only has a total of 256 MB of RAM.

[1] `https://github.com/tromp/cuckoo`

[2] The RAM available column for the Raspberry Pi family is based on a desktop based installation of Raspbian, which can be reduced. In fact, roughly 250 MB is occuped by Xorg and friends. The device did not run a Bitcoin node at the time, however.

[3] Including dedicated miners may seem strange, but (1) ASICs may be used by an attacker, and (2) ASICs may need to solve RAM dependent challenges in the future, which they currently cannot.

challenge size (MB)		
unoptim	optim	support
5	100	71%
32	640	57%
33+	660+	29%

Table 2: Support coverage over RAM required.

To facility optimized Cuckoo-Cycle with a 20x increase[4], the supported devices would change according to Table 2.

3 Hardware

Tests were run on the following devices:

- lubuntu on an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz with 16 GB of RAM and 8 MB cache per core.
- Raspberry Pi Model 3

All tests were executed on a single core.

4 Profiles

The two parameters being examined are:

- Size shift (in particular 20, 21, 25, and 28), a parameter which defines the size of the graph used in the cuckoo-cycle problem.
- Proof size range, a min and max for the number of edges in the cycle.

Among these, we investigate the correlation between proof ranges of the same size. (Is it more probable to find a 24-length cycle compared to a 48-length cycle in a random graph?)

[4] `https://github.com/xenoncat/cuckoo_pow`

4.1 Memory usage

We define the graph size over the number of nodes as

$$|Z| = 2^s \tag{1}$$

where s is the size shift parameter. The number of even nodes is half this value, i.e. $\frac{|Z|}{2}$, and from this the number of bytes occupied by edges becomes $\frac{|Z|}{16}$, and the bytes occupied by nodes becomes $\frac{|Z|}{8}$.

Consequently, the total amount of RAM as a function of the size shift s becomes

$$\frac{3|Z|}{16} = \frac{3 \cdot 2^s}{16} = 0.1875 \cdot 2^s \tag{2}$$

which means it effectively doubles for each size shift increment.

4.2 Size shift overview

size shift	~s/nonce	~nonces/s	factor	graph size
20	0.0377	26.5	-	192k
21	0.0801	12.5	2.13	384k
22	0.181	5.54	2.25	768k
23	0.383	2.61	2.12	1.5M
24	0.793	1.26	2.07	3.0M
25	1.63	0.612	2.06	6.0M
26	4.29	0.233	2.62	12M
27	17.7	0.0565	4.13	24M
28	44.8	0.00223	2.41	48M
29	97.8	0.0102	2.19	96M
30	176	0.00568	1.80	192M

Table 3: Nonce iteration time over size shift. The factor column shows the proportional (2.0 means double) increase in time per nonce compared to the previous size shift.

A size shift of 27 (Table 3) results in 18 seconds per nonce compared to 0.04 seconds for a size shift of 20. A significant increase can be observed at size shift 27, which has a factor 60% higher than the previous one. The RAM

used by the graph doubles (as expected) each size shift as can be seen in the rightmost column.

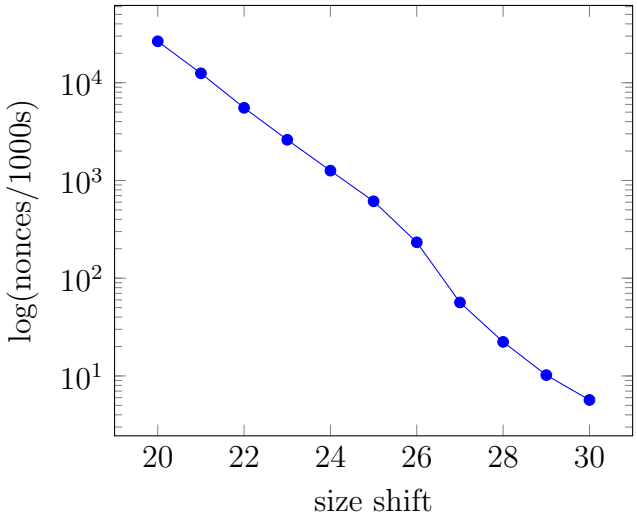

Figure 1: Nonces/1000s over size shift.

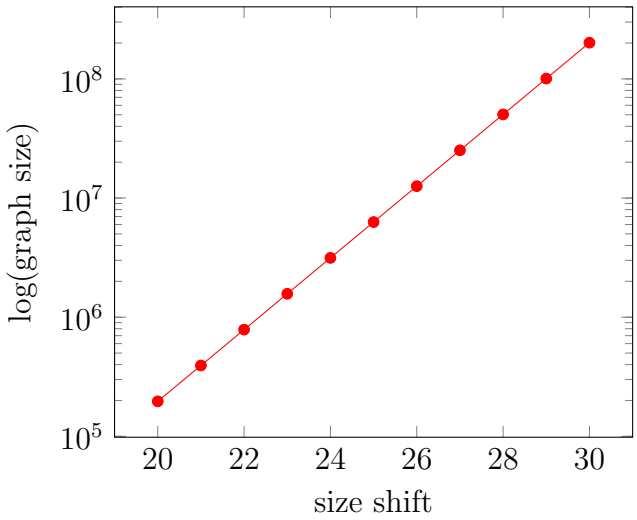

Figure 2: Graph size over size shift.

4.3 Proof ranges

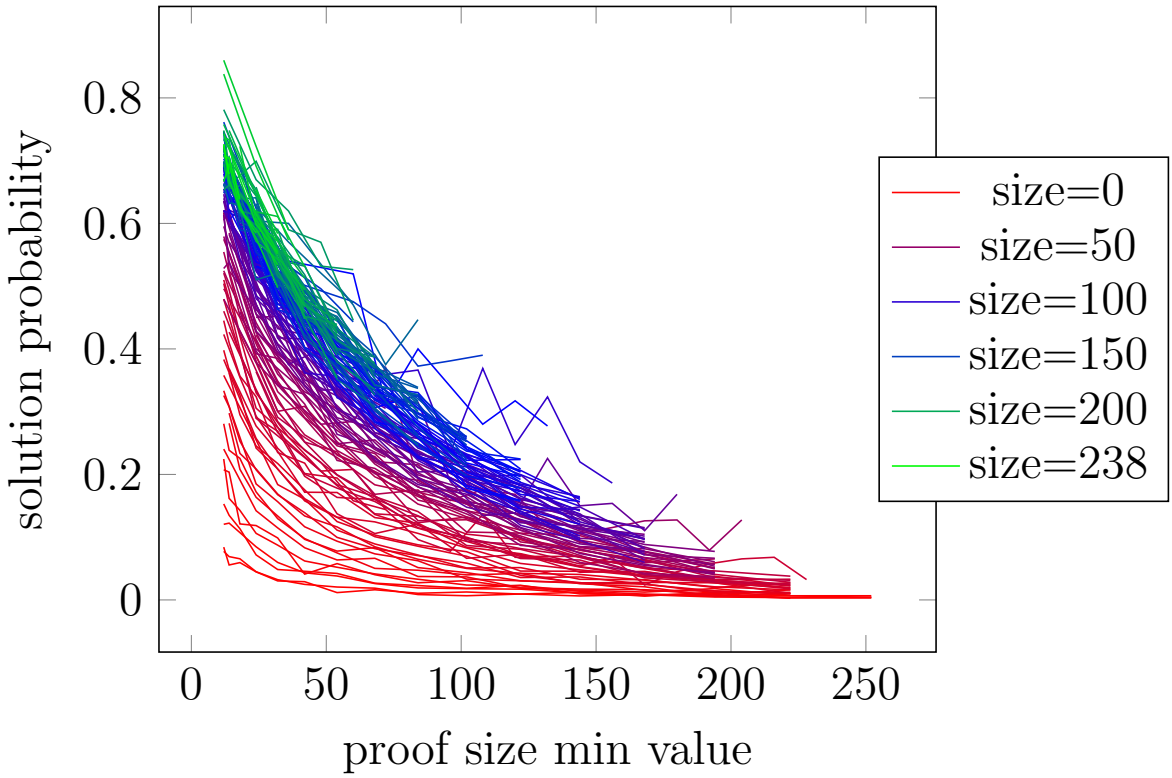

Figure 3: Solution probability (average solutions/nonce) as a function of proof size minimum over various sizes.

In Figure 3, the probability of finding a solution for a given nonce input is shown for various ranges (proof max minus proof min). The solution probability drops as the proof minimum increases, meaning that the difficulty of finding a solution is dependent on the proof minimum, regardless of the actual range. For instance, a zero-size (min=max) proof size [12..12] has a higher chance (8.1%) of finding a solution than a 34-size proof range of [144..177] (7.2%). The graph contains multiple size shift inputs, which all follow roughly the same path, implying that size shift is not a significant factor in solution probability[5].

[5]It is, of course, a significant factor in terms of *solution time* as the time to iterate over a nonce is dependent on the size shift (see Table 3).

4.4 Variance

The variance $s^2 = \frac{\sum(x_i - \bar{x})^2}{n-1}$ for the samples in the various parameters were analyzed, with the intention of finding a range that was predictable in terms of solution time. A low variance means high predictability. The results can be seen in Table 4 for sizeshift=28.

P min	P max	Size	Variance	P min	P max	Size	Variance
12	36	24	1.518431	12	60	48	2.179938
12	84	72	0.694444	12	108	96	0.864093
12	132	120	0.533199	12	156	144	0.473642
12	180	168	0.841858	12	204	192	0.868298
12	228	216	0.232011	12	252	240	0.254291
12	254	242	0.281790	24	48	24	4.967803
24	72	48	5.130904	24	96	72	3.090426
24	120	96	1.107345	24	144	120	1.711582
24	168	144	1.542532	24	192	168	1.529508
24	216	192	0.586659	24	240	216	0.434170
24	254	230	0.293706	36	60	24	9.608466
36	84	48	6.264706	36	108	72	3.175439
36	132	96	1.951691	36	156	120	1.336129
36	180	144	1.758431	36	204	168	1.039548
36	228	192	1.560491	36	252	216	1.250146
36	254	218	0.692212	48	72	24	21.664032
48	96	48	6.920635	48	120	72	3.435897
48	144	96	3.967366	48	240	192	1.081454
48	254	206	0.688136	60	84	24	8.961905
60	108	48	18.330049	60	132	72	4.213213
60	156	96	5.041963	60	180	120	1.840348
60	204	144	1.882535	60	228	168	1.374653
60	252	192	2.052525	60	254	194	1.673469
72	96	24	58.595833	72	120	48	4.548048
72	144	72	4.587702	72	168	96	1.840841
72	192	120	7.705645	72	216	144	2.389006
72	240	168	3.280702	72	254	182	1.345306
84	108	24	55.155556	84	132	48	15.115789

Table 4: Variance for sizeshift=28 case.

P min	P max	Size	Variance	P min	P max	Size	Variance
84	156	72	10.513369	84	180	96	4.758258
84	204	120	3.948718	84	228	144	4.708393
84	252	168	1.785990	84	254	170	7.854878
96	120	24	261.553571	96	144	48	44.229167
96	168	72	20.010526	96	192	96	14.543478
96	216	120	4.238859	96	240	144	7.112179
96	254	158	3.585825	108	132	24	35.016484
108	156	48	26.450292	108	180	72	62.417582
108	204	96	7.292319	108	228	120	6.105820
108	252	144	2.568151	108	254	146	4.243697
120	144	24	263.238095	120	168	48	27.232026
120	192	72	35.412088	120	216	96	5.558462
120	240	120	2.882576	120	254	134	8.147147
132	156	24	251.666667	132	180	48	54.265152
132	204	72	9.711462	132	228	96	8.210227
132	252	120	8.914021	132	254	122	8.653409
144	168	24	150.300000	144	192	48	65.174242
144	216	72	51.952381	144	240	96	20.545455
144	254	110	6.929885	156	180	24	156.194444
156	204	48	33.333333	156	228	72	26.000000
156	252	96	17.245614	156	254	98	7.223333
168	192	24	397.000000	168	216	48	69.923810
168	240	72	44.683824	168	254	86	41.450000
180	204	24	330.700000	180	228	48	19.307692
180	252	72	18.808824	180	254	74	53.666667
192	216	24	326.266667	192	240	48	83.982143
192	254	62	31.450000	204	228	24	53.238095
204	252	48	59.307692	204	254	50	59.151515
216	240	24	79.904762	216	254	38	66.265152
228	252	24	1242.966667	228	254	26	189.500000

Table 4 cont: Variance for sizeshift=28 case.

As expected, the variance decreases as the proof size range broadens, with a minimum $s^2 = 0.232$ observed at [12, 228]. It is clear that the variance decreases as the size of the range goes up, and discrepancies in this should be disregarded as noise. Nonetheless, $[12, 228]$ was picked as the default proof

size range; while $[12, 254]$ would have resulted in higher predictability, it was deemed negligible.

5 Results

This section describes the selected default parameters and the respective reasoning.

5.1 Size shift

A size shift of 28 was chosen as the default. This results in about 45 seconds per nonce. While this is very high, using appropriately large proof ranges should allow a solver to find a solution in a few iterations. The estimated RAM use ends up at 48 MB, which is slightly on the lower end.[6] As optimizations for Cuckoo Cycle are discovered, the default size shift will most likely increase in response, resulting in an expected increase in default memory requirements over time.

[6] With existing optimization methods, the 28 sizeshift challenge can be solved in a few seconds, but using significantly more RAM (around 900 MB). See `https://github.com/xenoncat/cuckoo_pow`.

5.2 Proof size range

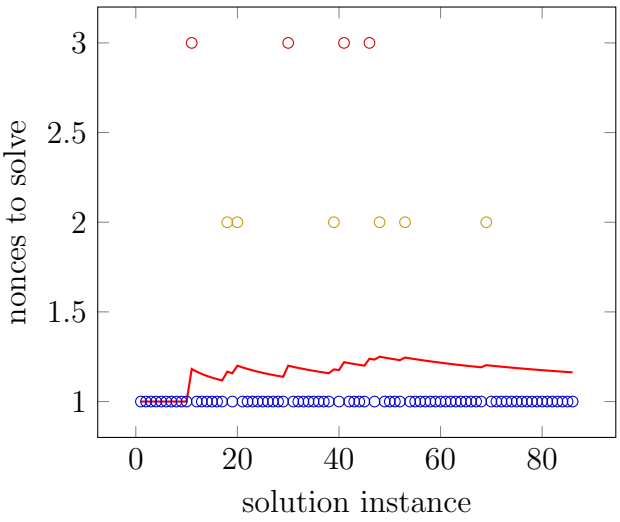

Figure 4: Nonces required to solve 86 challenges with size shift 28, proof size range [12..228].

A proof size range of [12..228] was chosen as the default. The aim was that it would take a normal node on average about one minute to solve the challenge, i.e. roughly $\frac{60}{45} \approx 1.33$ attempts. In Figure 3 on p. 7, this corresponds to the values around the $\frac{1}{1.33} = 0.75$ line. The average for the range [12..228] (size shift 28) was 1.16 nonces per solution. See Appendix A.

The longest solution time encountered was 3 nonces which would take roughly 2.3 minutes. 10 challenges (12%) were above the 1 nonce mark, meaning solution time is greater than 1.5 minute. The bulk of the challenges were solved in one try, i.e. 45 seconds. See Figure 4.

A Sizeshift 28 proof size range [12..228]

```
# [prf rng]     solved  nonces  n/sol   cycles          time
# 12    228     86      100     1.16    14775115174872  4335.449250

# [prf rng]     nonces
12      228     1
12      228     1
```

12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	3
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	2
12	228	1
12	228	2
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	3
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	2
12	228	1

12	228	3
12	228	1
12	228	1
12	228	1
12	228	1
12	228	3
12	228	1
12	228	2
12	228	1
12	228	1
12	228	1
12	228	1
12	228	2
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	2
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1

12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1
12	228	1