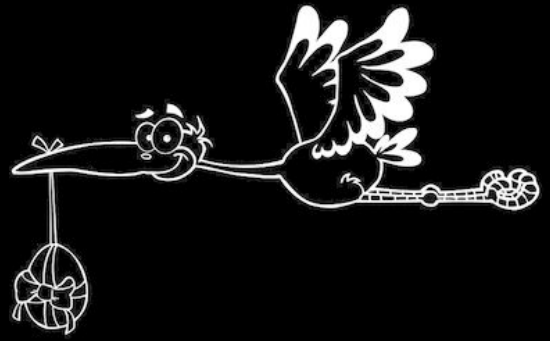
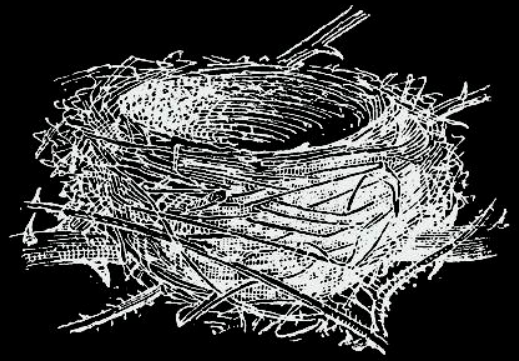
grincon1
2019.11.22 // c-base berlin



One Flew Over the Cuckoo's Nest



QTUM

SPARK POOL

Grin’s dual PoW

Five stages of PoW grief

- [**Denial**] CPU only / GPU hostile
- [**Bargaining**] GPU friendly / ASIC resistant
- [**Anger**] Need to hardfork to brick the ASICs
- [**Depression**] 6-month PoW tweaks to disincentivize ASICs
- [**Acceptance**] Embrace ASICs
    - on last PoW that failed to resist them [Litecoin, Dash, ZCash]
    - on a new PoW that’s friendly to them [Ethereum Classic, Monero?!]

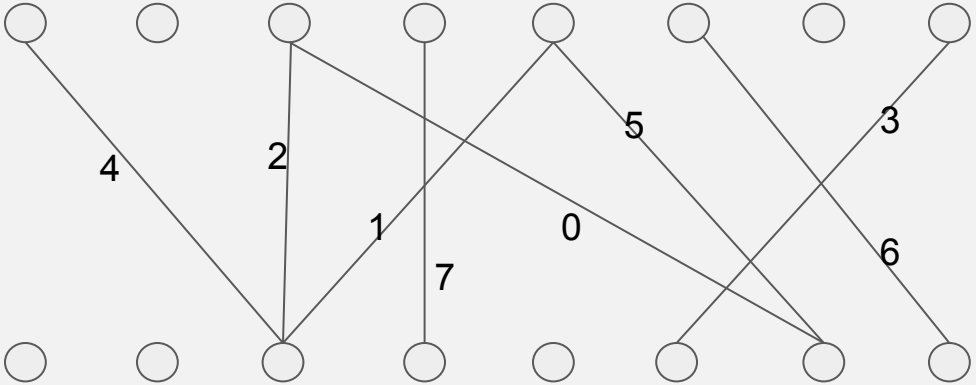
Grin's Dual PoW

- Smoothly transition from ASIC resistant to ASIC friendly over 2 years
- Resistant: CuckARoo*29
  - 6 month tweaks leave little or no time to ROI
  - Computing edge endpoints in blocks of 64 edges penalize lean solving
  - Memory bandwidth requirements raise design complexity
- Friendly: CuckAToo31+ with 2020 transition to Cuckatoo32+
  - C32 is essentially Proof-of-512MB-SRAM
  - Might fit in CPU cache in a few decades

CuckARoos

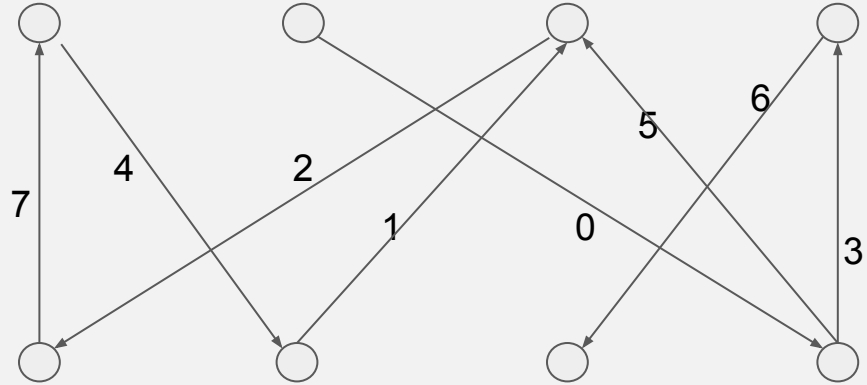
Cuckaroo

N undirected edges between N + N nodes



Cuckarood

N + N directed edges between N + N nodes



Cuckarood

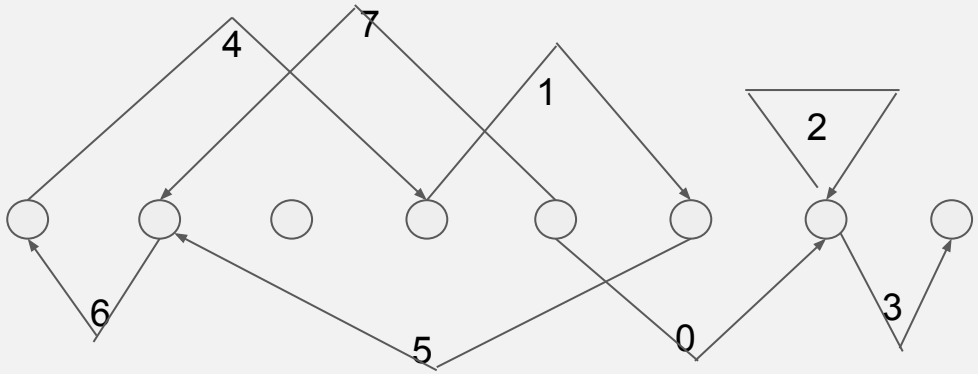
```
template <int rotE = 21>
class siphash_state {
 void sip_round() {
   v0 += v1; v2 += v3; v1 = rotl(v1,13);
   v3 = rotl(v3,16); v1 ^= v0; v3 ^= v2;
   v0 = rotl(v0,32); v2 += v1; v0 += v3;
   v1 = rotl(v1,17); v3 = rotl(v3,rotE);
   v1 ^= v2; v3 ^= v0; v2 = rotl(v2,32);
 }
}

siphash_state<25> shs(keys);
```

CuckARooM

M is for Monopartite

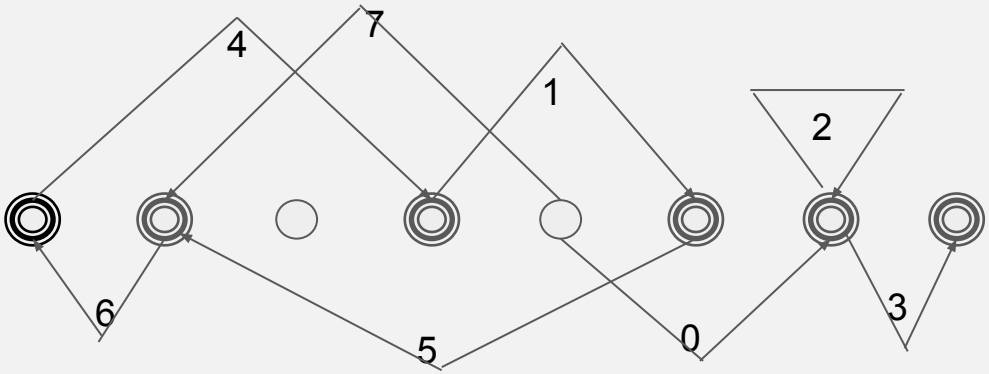
Cuckaroom

N directed edges between N nodes



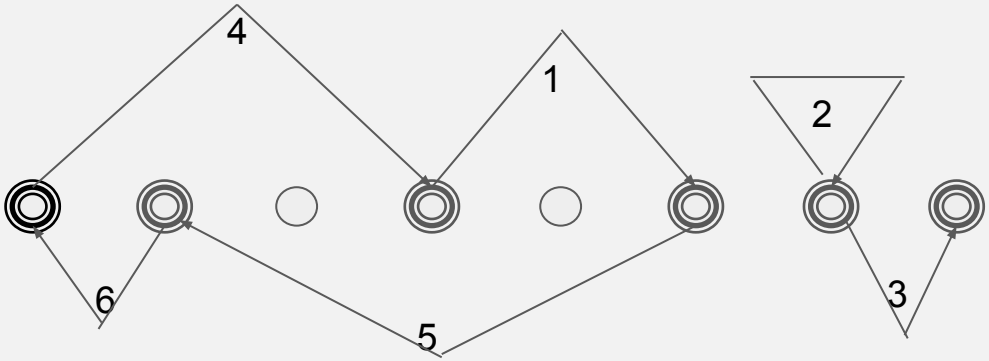
Cuckaroom

1a. Mark endpoints of edges



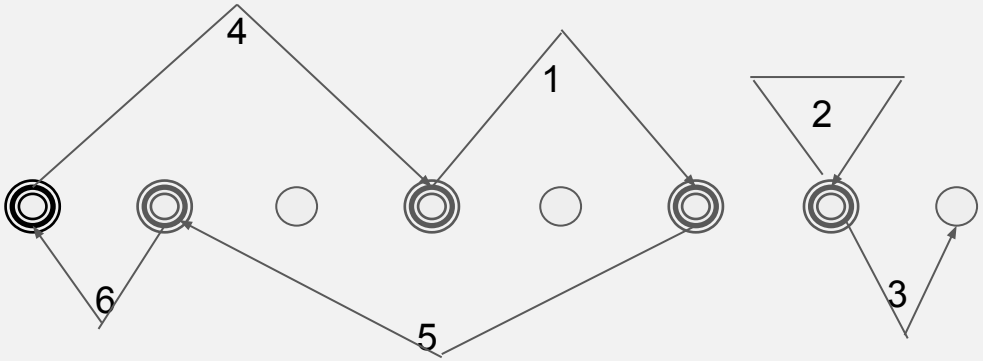
Cuckaroom

1b. Kill edges with unmarked startpoint



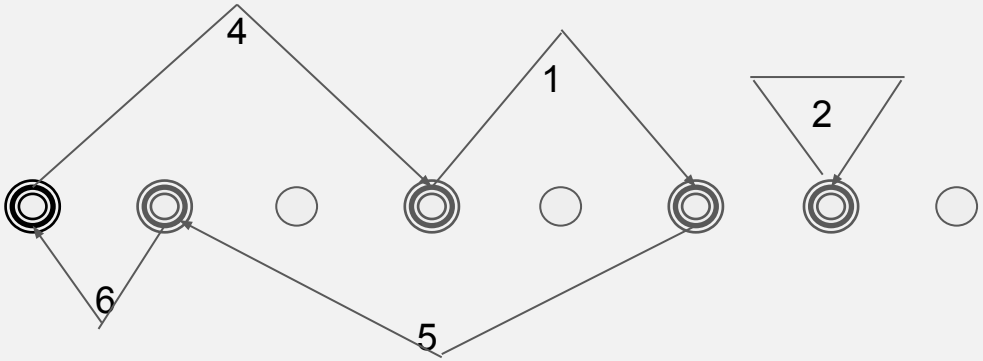
Cuckaroom

2a. Mark startpoints of edges



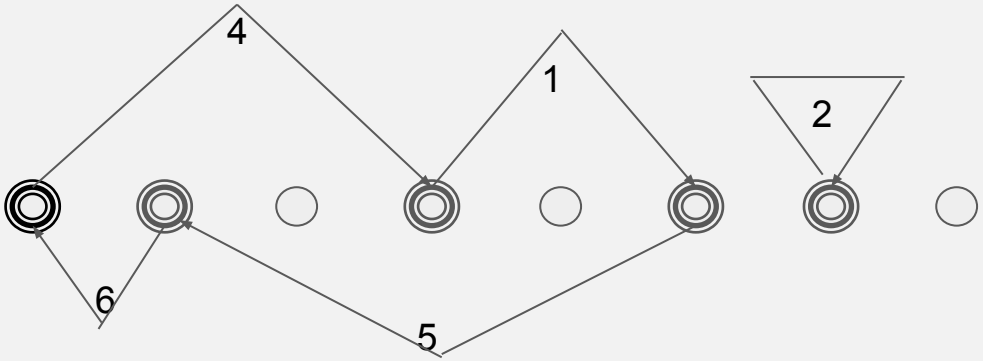
Cuckaroom

2b. Kill edges with unmarked endpoint



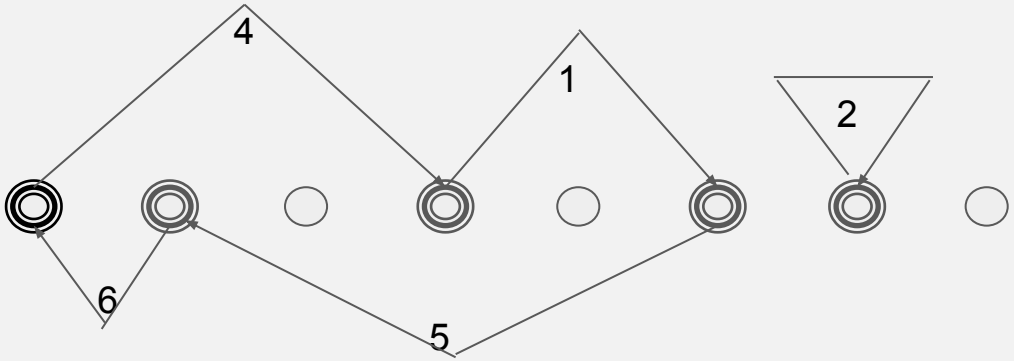
Cuckaroom

After many rounds one is left with only cycles



Cuckaroom

Expected number of L-cycles



N^L sequences i_1 i_2 .... i_L
Prob. (1/N)^L of forming cycle

=> Expected number of cycle forming sequences ~ 1

L-cycle has L such sequences

4 1 5 6
1 5 6 4
5 6 4 1
6 4 1 5

~ 1/L

Cuckaroom

```
const u64 last = buf[EDGE_BLOCK_MASK];
for (u32 i=0; i < EDGE_BLOCK_MASK; i++)
  buf[i] ^= last;
```

```
for (u32 i=EDGE_BLOCK_MASK; i; i--)
  buf[i-1] ^= buf[i];
```

Summary

Summary

- Cuckaroom uses Monopartite graphs, including odd cycles
- has directed edges like Cuckarood
- uses standard siphash like Cuckaroo
- computes edge blocks differently, xoring ALL later states
- expects 1/42 solutions per graph, half of Cuckarood
- reference CUDA solver uses global node bitmap
- needs twice as many rounds to trim give fraction of edges