Methods of inclusion:

shift (Eq: 93)
insert (Ez: 73)
append
[C: 3, h: 10, a: 1 ...]
What method is most friendly toward least amount of rehashes?

constant size
Sha3
|
Sha3
|
constant size
Sha3
|
Accumulator of data
Grows with # of users
|
Submit (Hash prev, Hash next, State prev, Transfer[] txs)
|
a: 1
b: 2
⋮
z: 7
|
Can cause a temporal bandwith backlog, e.g. 5 tx/s
|
a → b: 1
c → d: 3
e → z: 5
|
apply (Transfer[] txs, State prev) == Hash next ✓

Merkle Proof

Batch #1 0xa
Batch #2 0xb
0xc
"Batch chain"

Consolidation

out out out in
Can we design an incentive for it?
- tx fee: bytes/ETH

ERC20 Rollup ERC20
"Spent" out in
"Unspent" out X
Handling deposits and withdrawals becomes easier.

thinking in UTXO is easier
out in UTXO?
or
a: 1
b: 2
Account-based?
thinking in accounts is more interesting

H(...)
H(...) H(...)
H(...) H(...) H(...) H(...)
0x1: 0,5 0x2: 3 0x3: 2 0x4: 7
Order criteria are similar to accounts: In what way is the least amount of rehashing necessary?

Control over
utxo.filter(o => o.val < 0.5)
size of utxo tree
activity filter?
active | inactive

Highest balance: d: 15, a: 13, z: 9, e: 1
Last access: 1s f: 2, 2min z: 5, 3min b: 3, 1h c: 13
Lexical and reverse: a: 7, b: 5, c: 3, d: 2, e: 13
Lowest Balance: g: 0, m: 0, L: 1, b: 3

H(...) → Binary? Sparse?
H(...) H(...)
H(...) H(...) H(...) H(...)
a: 1 b: 1 c: 1 d: 1
Criteria for establishing order:
- As little rehashing as possible
- Order should NOT move leaves
- Order can only arise though implicit data e.g. balance but not eg. transfer frequency.

May 21, 2021