RED QUEEN'S SYNC PROTOCOL FOR ETHEREUM

ANDREW ASHIKHMIN & ALEXEY AKHUNOV

ABSTRACT. TODO: abstract.

"A slow sort of country!" said the Queen. "Now, here, you see, it takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"

Lewis Carroll, Through the Looking-Glass and What Alice Found There

1. Introduction

TODO: mention the sync failure problem Akhunov [2019a] and the needs of light clients like Mustekala. Inspirations like BitTorrent, Parity's warp sync. Snapshot synchronisation rather than from genesis.

2. Notation

We mostly follow the conventions and notations of the Yellow Paper Wood [2018], for instance $\mathbb{Y}$ denotes the set of nibble sequences. We use the letter π for prefixes of state or storage trie keys $\mathbf{k} \in \mathbb{B}_{32}$,

$$\pi \in \mathbb{Y} \ \land \ ||\pi|| \leq 64 \tag{1}$$

A key matches a prefix iff all their first nibbles are the same,

$$\texttt{MATCH}(\mathbf{k}, \pi) \equiv \forall_{i<||\pi||} : \mathbf{k}'[i] = \pi[i] \tag{2}$$

($\mathbf{k}'$ is a sequence of nibbles, while $\mathbf{k}$ is a sequence of bytes.)

3. Protocol Specification

TODO: protocol spec; cross-check it against geth's fast sync spec.
o – reply overhead in bytes.
b – size of a branch node in bytes.
l – average leaf size in bytes.
$||R_b||$ – number of nodes in reply.
$||R_l||$ – number of leaves in reply.
Thus the size of a reply, assuming the average leaf size, is

$$S(R) = o + ||R_b||b + ||R_l||l \tag{3}$$

Date: March 2019.

TODO: RLP changes the formula slightly.

4. Suggested Full Sync Algorithm

TODO: top-level trie with branch nodes only; nodes track blocks; phase 1, phase 2.

5. Performance Analysis

For this analysis we assume that all tries are well balanced. We also assume that all top nodes up to a certain trie level i are branch nodes, not leaf nor extension nodes. This is a reasonable assumption if i is not too big—see Akhunov [2019b].

t – total number of leaves in the trie.

TODO: optimal phase 1 depth.

Now let us find the maximum size d of the request prefix π that makes sense to use when we are catching up (phase 2 of the sync). Let assume that only one leaf that matches π has changed. (If we know that there are no changes, there is no need for a sync request.) That is a reasonable assumption if we are not too many blocks behind, there are not that many changes per block, and d is sufficiently large. (For instance, if we are 100 blocks behind, and there are 500 leaf changes per block, then $d \geq 4$ will suffice.) Consider two options: request the prefix π or send requests with prefixes $\pi \cdot 0, ..., \pi \cdot 15$ of size $d+1$ (not necesseraly all 16 of them). In the first case we receive a reply of the size, on average,

$$S = o + ||R_b||b + \frac{t}{16^d}l \tag{4}$$

For the second case we need to send at most two requests, as the first reply will give us the information to identify which nibble has changed. (With the probability $\frac{1}{16}$ the second request is not necessary.) The combined size of those 1 or 2 replies, on average, is

$$S' = \left(1 + \frac{15}{16}\right)o + (||R_b|| + 1)b + \frac{t}{16^{d+1}}l \tag{5}$$

It does not make sense to prefer requests with longer prefixes if $S \leq S'$. Solving this inequality, we obtain

$$16^d(16b + 15o) \geq 15tl \tag{6}$$

In other words, it does not make sense to use prefixes longer than

$$\left\lceil \log_{16} \frac{15tl}{16b + 15o} \right\rceil \tag{7}$$

TODO: convergence analysis.

6. Conclusion

TODO: conclusion.

References

Alexey Akhunov. Looking back at the Ethereum 1x workshop 26–28.01.2019 (part 1), January 2019a. URL `https://medium.com/@akhounov/looking-back-at-the-ethereum-1x-workshop-26-28-01-2019-part-1-70c1ebd93266`.

Alexey Akhunov. Looking back at the Ethereum 1x workshop 26–28.01.2019 (part 2), February 2019b. URL `https://medium.com/@akhounov/looking-back-at-the-ethereum-1x-workshop-26-28-01-2019-part-2-d3d8fdcede10`.

Gavin Wood. Ethereum: A Secure Decentralised Generalised Transaction Ledger, December 2018. URL `https://github.com/ethereum/yellowpaper`.