Matrix 区块链分片方案设计思路

# 总体设想

## 交易分区

每个交易首先进行预交易验证，生成状态变化表

Address0: state0

Address1: state1

Address2: state2 …

根据状态变化表，把交易分区放置(1对多)

Range[Address0],Range[Address1],Range[Address2]…

## 状态分区

每个Range的交易跑完后，该Range的状态树根与区块头的状态树根一致[所有相关状态的交易都有了]

多个Range可以在同一节点验证

## Sharding

验证者分组：同交易分区

交易分发网络：

分组验证与状态树合并

交易分组：

交易有n个关联账户状态变化，则交易有n位状态变化指示数组

Tx.stateMark []byte

每个区域验证自己的状态树，如果验证成功，则状态指示标记为1，否则标记为0

总体验证：

只有所有状态位为1的交易才是合格的交易

# 区块数据结构

## 区块头数据

ROOT（状态树）

状态树层次为

总Merkle树，256个子Merkle树

交易hash

Receipt Hash

# 交易数据

1. 区块链数据分层，按照账户的地址分组
   1. 账户地址首位byte，分成256个组，每个组有自己的账户信息、交易信息，receipt信息，每个区块高度把当前的账户更新，交易，receipt的交易数据包打包为一个数据包文件，每个数据包有一个hash。
   2. 256个数据包有一个总hash，以及第二层的hash，上区块链的header
2. 区块链数据下载为按需下载，validator等顶层节点为全下载
   1. 节点A有n个账户，n个账户属于上面的m个分组，m<=n
   2. 分组0为区块链公有数据信息，每个节点都要下载
   3. 节点A需要下载分组0+m个分组，以及所有分组的hash
3. 每个节点都参与维护区块链的安全
   1. 每个节点下载自己的分组，验证分组hash以及总hash
   2. 每个节点下载m个分组，保证m个分组的安全
   3. 多个节点随机验证不同的分组，足够多的节点可验证全部数据
4. 优点
   1. 本地账户可查询账户状态，余额等
   2. 本地账户可查询自己发送的交易
   3. 本地存储数据量为m/256的量级
5. 下载方法参考IPFS，目前正在研究，或者用目前以太坊的下载也可以
   1. IPFS的优势就是他有完整的网络数据定位，知道去哪些机器下载，以太坊的下载目前没有这个机制