区块链分布式存储模式设计

目标

结合纠删码以及局部修复码技术，提出一种基于纠删码的区块链系统区块文件存储模型， 用于保证数据修复能力的前提下提高区块链节点的存储效率。

相关内容

区块链系统

现有区块链存储方式中，每个区块链节点都有一份完整的副本数据（最长链），同时节点也都拥有独立计算能力。所有处于同一区块链网络中的节点都需遵循共识协议，以保证各节点计算存储的数据是相同的，包括数据区块的顺序，内容等。同时，很重要的一点是， 若区块链网络中有部分节点（少量）失效或者离开网络，其他节点依然能够通过已有数据进行数据校验以及数据恢复等操作。

纠删码

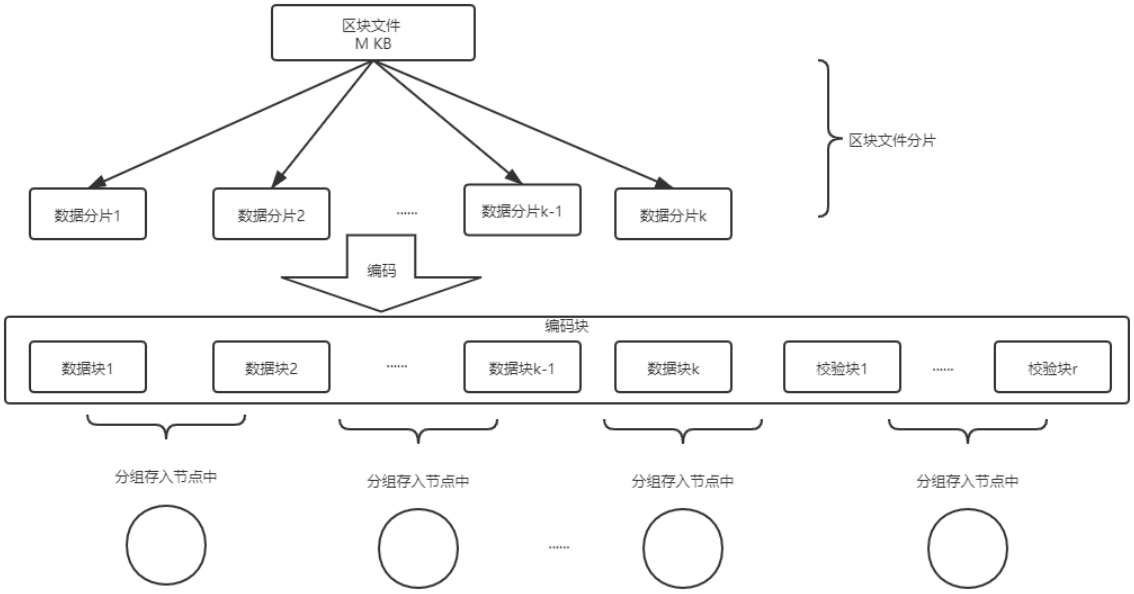
以RS码为例， 该码可将k份原始数据通过增加m份校验数据的方式，以保证在任意不大于m份数据失效丢失的情况下，还能够通过剩余的数据完整恢复出原始数据。

区块链存储结构

1. 区块：链上保存交易信息的存储单元
    1. 区块头：哈希值
    2. 区块体：交易信息
2. 节点
    1. 轻节点
    2. 全节点
    3. peer节点（超级账本）
        1. 存放全量数据
        2. 存放Channel的数据
    4. order节点（超级账本）
        1. 可以通过order节点控制信息的传播
        2. 提供可插拔共识服务
        3. 保存Channel的配置（System chain）

基于纠删码的区块链文件存储模型

区块链节点将其保存的完整区块文件通过纠删码技术编码成多个编码块，每个节点仅保留部分编码块，全网节点拥有完整的编码块信息，使得全网中各个节点在尽可能减少存储空间占用的同时，又能恢复出完整的区块文件。



模型一：

编码(RS码)存储过程

本文采用一种基于通信的方案，在超级账本区块链网络搭建完成初期产生第一个区块文件之前，排序服务集群会确认加入网络的组织的先后顺序，当有新的组织加入该网络或者退出该网络时，排序服务集群都会定期向其余组织的背书节点广播编码控制信息，包括用于指示切分区块文件的原始块数量、纠删码编码算法、编码容错率、节点数量以及**编码块组号与节点的对应关系。**

1. 第一个区块文件(规定区块文件大小为m如128kB)完成后，所有记账节点将根据最近一次接收到的编码控制信息对区块文件进行纠删码编码存储，每当有节点加入或者离开时，更新编码控制信息并重新广播给各节点。
2. 在一个完整区块文件还未生成时，区块信息仍然采用原有的存储方式进行副本存储。

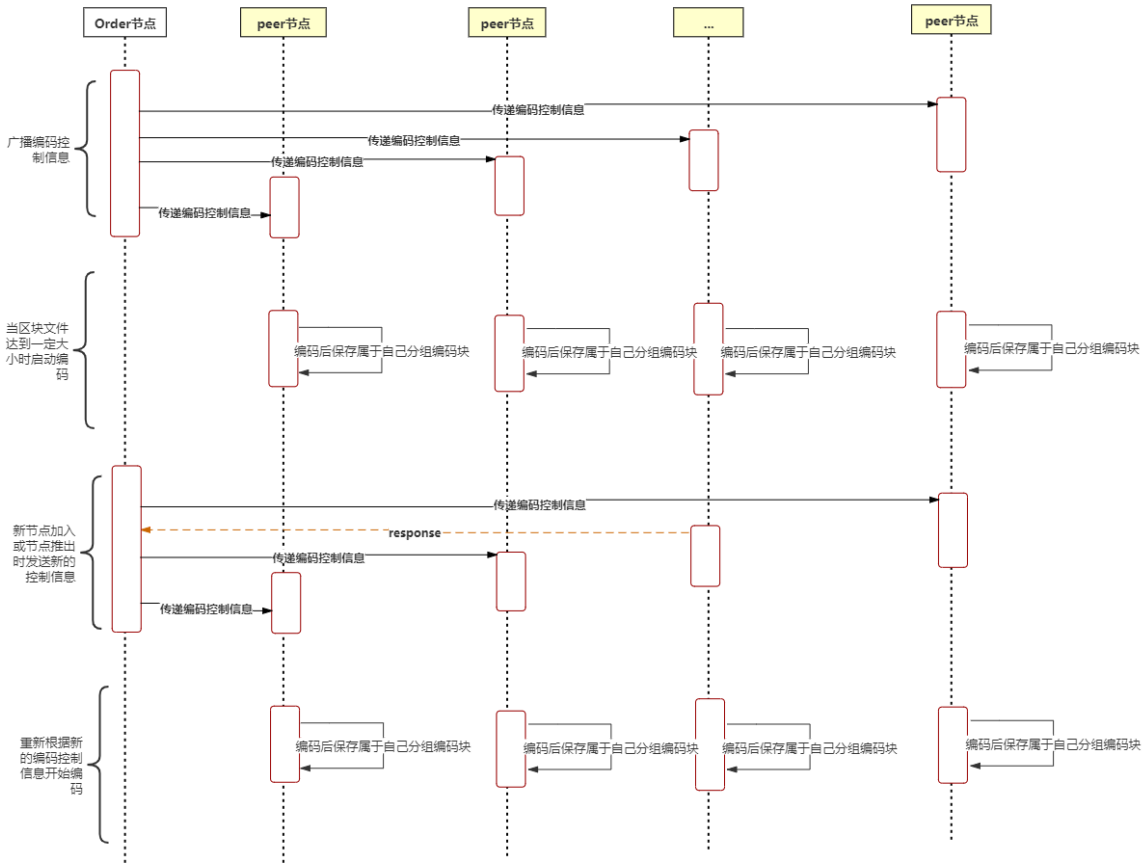
纠删码容错率λ，原始文件切分为k个数据块，$r = k\frac{\lambda}{1-\lambda}$个校验块，$t$个节点(组织)分发$(k+r) = k\frac{1}{1-\lambda}$编码块。每个节点保存$\frac{k+r}{t} = \frac{k}{(1-\lambda)t}$个编码块的数量，由编码控制信息确定保存第$i$组编码块。

$$每个节点保存的编码块数量：n_{node} = \frac{k+r}{t} = \frac{k}{(1-\lambda)t}$$

$$每个节点保存的编码块大小：m_{node} = \frac{m}{k}n_{node} = \frac{m}{(1-\lambda)t}$$

$$全网节点保存编码总大小：m_{total} = m_{node} \times t = \frac{m}{1-\lambda}$$

编码存储过程



解码恢复过程

1. 客户端向某个节点发起区块链查询请求
2. 节点向Orders节点发起编码块请求
3. Orders节点向已连接的peer节点广播该节点
4. 向请求节点传输编码块

编码状态下的同步更新方法

通过发送节点账本快照判断邻近节点是否需要发送更新给同步请求发起节点。

1. 邻近节点为未编码节点：直接将区块信息返回给发起同步请求节点
2. 邻近节点为编码节点：通过解码恢复方法恢复区块数据后发送

模型二

目标

在节点加入或离开不需要order节点进行编码通知信息的广播，通过个节点的公钥证书相关属性，在节点加入时就可知该节点所属分组。

编码(RS码)存储过程

1. 和模型一中相同，在一个区块文件还为生成时，节点都采用副本方式存储。同时，在区块文件生成**根据公钥哈希模q所得值进行分组**
2. 组内通过广播的方式进行组内节点进行纠删码编码存储。

设组数为q， 组内节点数为$n_0, n_1, \ldots, n_{q-1}$, 第$i$ 组每个节点所需保存的编码块个数以及编码数据大小为, 其中$\lambda^{'}$为组内容错率：

$$
\begin{aligned}
&\text{第}i\text{组各节点编码块个数}：n_{node} = \frac{k+r}{n_i} = \frac{k}{(1-\lambda^{'})n_i} \\
&\text{第}i\text{组各节点保存编码块大小}：m_{node} = \frac{m}{k}n_{node} = \frac{m}{(1-\lambda^{'})n_i} \\
&\text{其中} \quad i \in [0, q-1], \lambda^{'} = 1 - \sqrt[q]{1-\lambda}
\end{aligned}
$$

$$
\text{全网节点保存编码总大小}：m_{total} = \sum_{i=0}^{q-1} \frac{m}{(1-\lambda^{'})n_i} \times n_i = \frac{q \times m}{1-\lambda^{'}} = m \times \frac{q}{\sqrt[q]{1-\lambda}}
$$

解码恢复过程

同模式一，但模式二中仅需要向组长节点获取解码所需节点组内解码即可

同步更新

同模式一方式一致，仅将解码恢复方式同理换成模式二解码恢复方式。

需要解决的问题：

1. 模式二中能否不采用组长节点广播的方式，让节点可以自行计算出所需保存的编码块组
2. 通过编码实验的方式确定性能较佳的参数