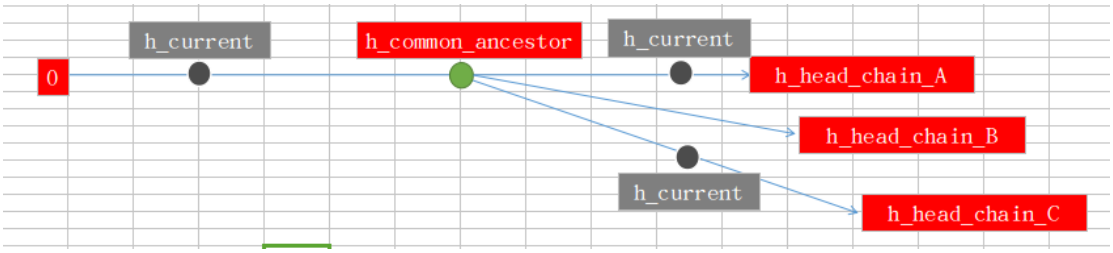
同步方案分析与设计

作者：何正军

日期：2018.2.6

当区块链网络中有新节点加入或者重启之后，或者其他一些原因导致本节点的块高度低于其他节点的高度时，就需要进行区块的同步。但是考虑到从其他单个节点进行同步会导致对端节点的网络高负载，而从多个节点进行同步时，又会由于分叉的情况，导致部分区块不能顺利的进行链接，于是需要分析存在的可能情况，进行一些实现上必要的设计，使得我们的同步机制比较高效、稳定、可靠。

说明：以下基本上是从 GETH 项目的实现方案分析，我们可以从中借鉴一些思路，我们可以在实用性和开发工作量之间进行平衡。



如上图所示，定义如下：

h_current:本节点需要开始同步的高度；

h_common_ancestor:区块链网络中存在的分叉开始的块高度；

h_head_chain_A, h_head_chain_B, h_head_chain_C 分别是不同 chain 中的最新高度，同时满足要求：

h_head_chain_C > h_head_chain_B > h_head_chain_A；

因为我们假设当前最大工作量链接就是 chain_C，在发起同步前，需要确保和我们同步的 chain 不存在分叉，否则需要回退到分叉开始节点进行同步。

（一）寻找开始同步节点

◆ 场景一：处在分叉分支上

在开始同步之前，我们需要确认当前最新节点是否与主链存在分叉，因为如果存在分叉，则需要从分叉开始之处进行同步，避免出现同步过来的块成为孤块，而又不能同步之前的块，而导致同步失败。

这样的场景类似下面的情况：

当我们的 chain 处在 chain_B 上的分支时，且 h_current > h_common_ancestor，此时若直接从 chain_c 的相同高度开始同步，必然导致同步过来的新块不能与当前链路不能链接，需要向后寻找公共父块，然后从该公共父块向后同步；

◆ 场景二：处在分叉后相同分支上

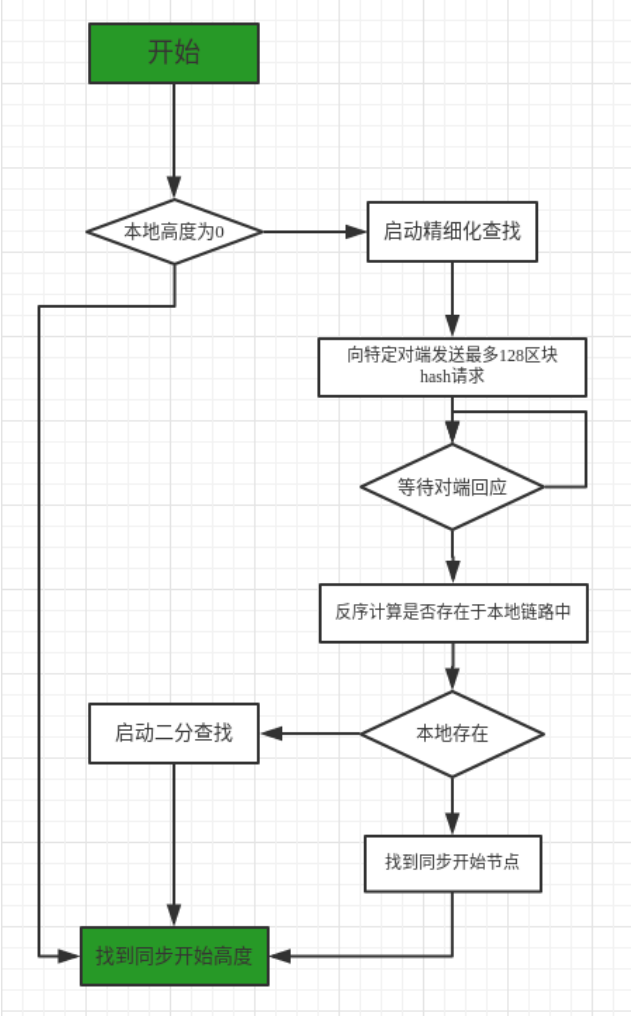
如果本来就处在相同的分叉后的相同分支上，则可以从当前高度向后直接进行同步。

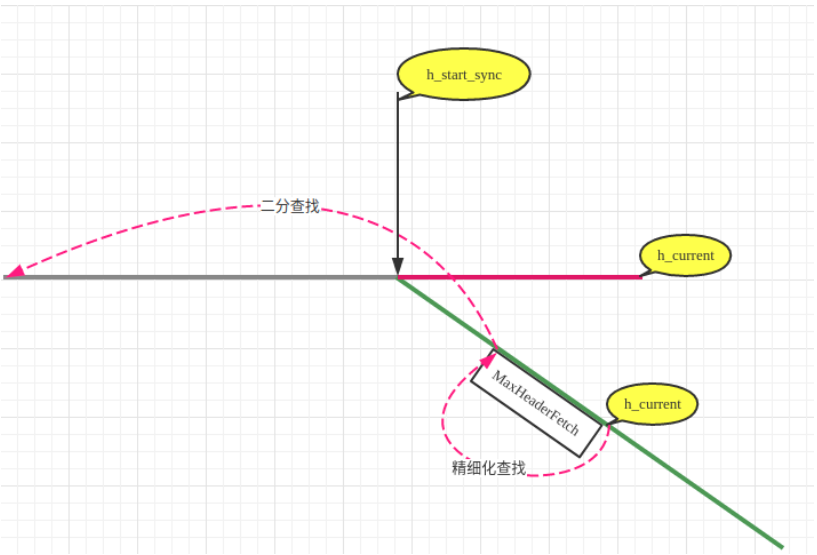
◆ 场景三：处在分叉前的主干分支上

如果当前节点的当前高度处在网络中分叉开始之前的高度上，则可以从当前高度开始直接同步，只是需要保证分叉之后的块数据是从 chain_c 上进行同步的就可以，因为当前节点会从保持连接的所有对端节点进行同步，我们需要确保进行同步的数据块是我们想要的数据块，而不是不同分支上的数据块，于是我们需要先要从 chain_C 上获取相应的块的 hash。[hash_start, height_start:hash_end, height_end],只要保证在向请求同步的对端接收到该起始块的高度和 hash 值之后，确认存在就可以发送数据块，否则这些数据块同步失败。需要从特定的对端进行同步（如发现最大高度的对端）。

◆ 开始同步父块高度寻找算法

我们已经介绍了处在不同高度情况下进行同步的三种场景，但是我们需要确定统一的开始进行同步的父块高度寻找算法，如前面三种场景描述的那样，最后确定的开始同步高度会小于或等于 h_current。具体寻找算法描述如下：





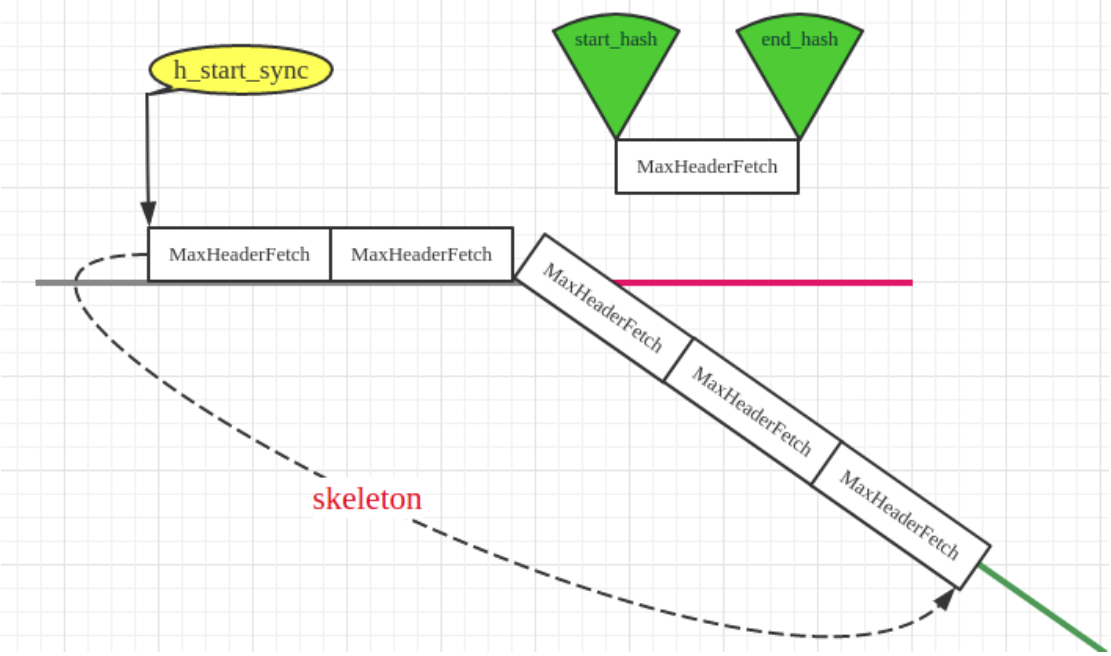
① 如果本节点当前高度为 0，则直接确定为从 0 开始同步；

② 如果 h_current > MaxHeaderFetch(128)，则首先启动精细化查找算法。精细化查找算法会反向查找深度为 MaxHeaderFetch 块头的 hash 值与高度，等到接

收到区块头信息，进行反序确认，一旦找到就认为该高度就是需要开始同步的高度，

③ 如果精细化查找失败，即没有在 MaxHeaderFetch 范围内找到开始同步的节点，则启动二分查找，最后一定会找到开始同步的区块高度，否则就将开始同步高度设置为 0；

（二）区块同步机制



① 采用负载均衡机制

我们找到了开始同步的高度值定义为：h_start_sync，此时可以开始同步了。因为如果仅从单个 peer 进行同步的话，会导致该对端的网络吞吐急剧增加，影响其正常的网络收发需求，所以我们采用同步负载均衡的策略机制。

一次从一个对端最多请求 MaxHeaderFetch 个区块的同步请求；

② 生成区块链路骨架信息

为了避免存在分叉情况下，甚至多个分叉的情况，同时分叉长度比较大的情况，我们会首先从最大工作量分支获取其骨架(skeleton)信息，并通过包含一段骨架信息的区块请求发送给其他 peer，如果对端能够在本地找到该请求所需要的区块，则发送结果；否则返回不存在的信息。

如果本地接收到对端不存在的错误信息，则直接从最大工作量节点进行请求；

其中的骨架信息生成机制是将 h_start_sync 开始的区块每隔 MaxHeaderFetch 将对应的首尾区块的 hash 作为局部区块链请求的信息。

③ **最后少于 MaxHeaderFetch 的区块直接从特定 peer 进行同步**

因为在通常情况下，如果存在分叉，一般都是发生在区块链的最近刚生成的分支上，所以为了避免从不同分支的节点同步失败的可能，该部分的同步仅仅从发现的最大工作量分支节点进行同步。