0到1全面认知波卡——自我升级（八）

原创 可达鸭Joie 鸭说区块链 2020-08-05 15:51

往期回顾：

0到1全面认知波卡——概述（一）
0到1全面认知波卡——跨链可组合性（二）
0到1全面认知波卡——异构分片（三）
0到1全面认知波卡——共享安全（四）
0到1全面认知波卡——提名权益证明（五）
0到1全面认知波卡——一键发链（六）
0到1全面认知波卡——平行线程（七）

上回学习了**平行线程**的概念，平行线程是波卡的平行链划分的很多并行运行的线程，项目方只需很少的押金和费用，就可以享受到和完整平行链一样的功能和共享安全。

平行线程意味着项目方可以根据需求进行上链，对上链频次不高的项目提供了最具性价比的方案，从而使整个波卡的生态不仅能出现巨头项目也能包容无数个小项目，使波卡真正意义上成为区块链的**最具普适性的底层基础协议。**

今天学习波卡的重要概念——**自我升级。**

在互联网世界中软件升级是一件习以为常的事情，通常只需要下载覆盖，这样就可以修复错误或者享受到产品最新的功能。

然而，在区块链的世界中，升级是一件非常麻烦的事情，搞不好可能会出现有争议的硬分叉，分裂整个社区。

分叉是每个区块链都必须要面对的问题，什么是分叉？如何避免分叉？波卡又是如何做到永远不会分叉？

此文将会是本科普系列中最难理解的一篇，鸭哥将会带大家慢慢解读。

一、什么是分叉

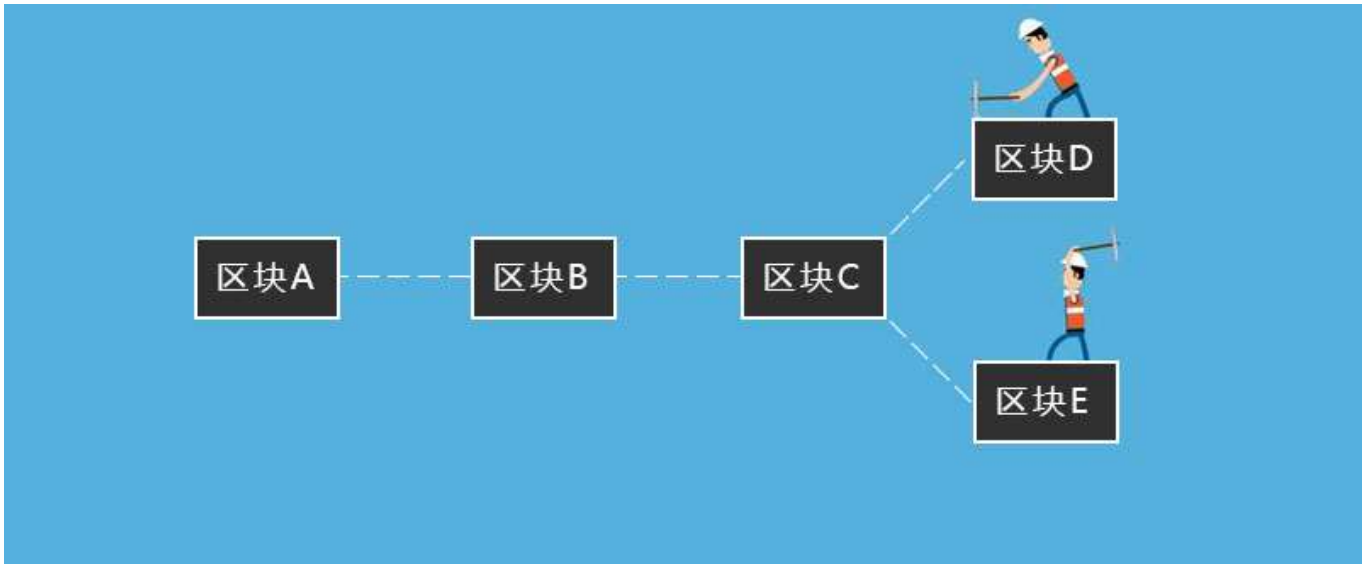
以比特币为例。

比特币大概每10分钟会产生一个新的区块，节点需要通过解数学题的方式争夺这个区块的记账权，当有一个节点赢得记账权时，节点会把这10分钟内的交易记录打包进区块内，并广播给附近的节点同步。

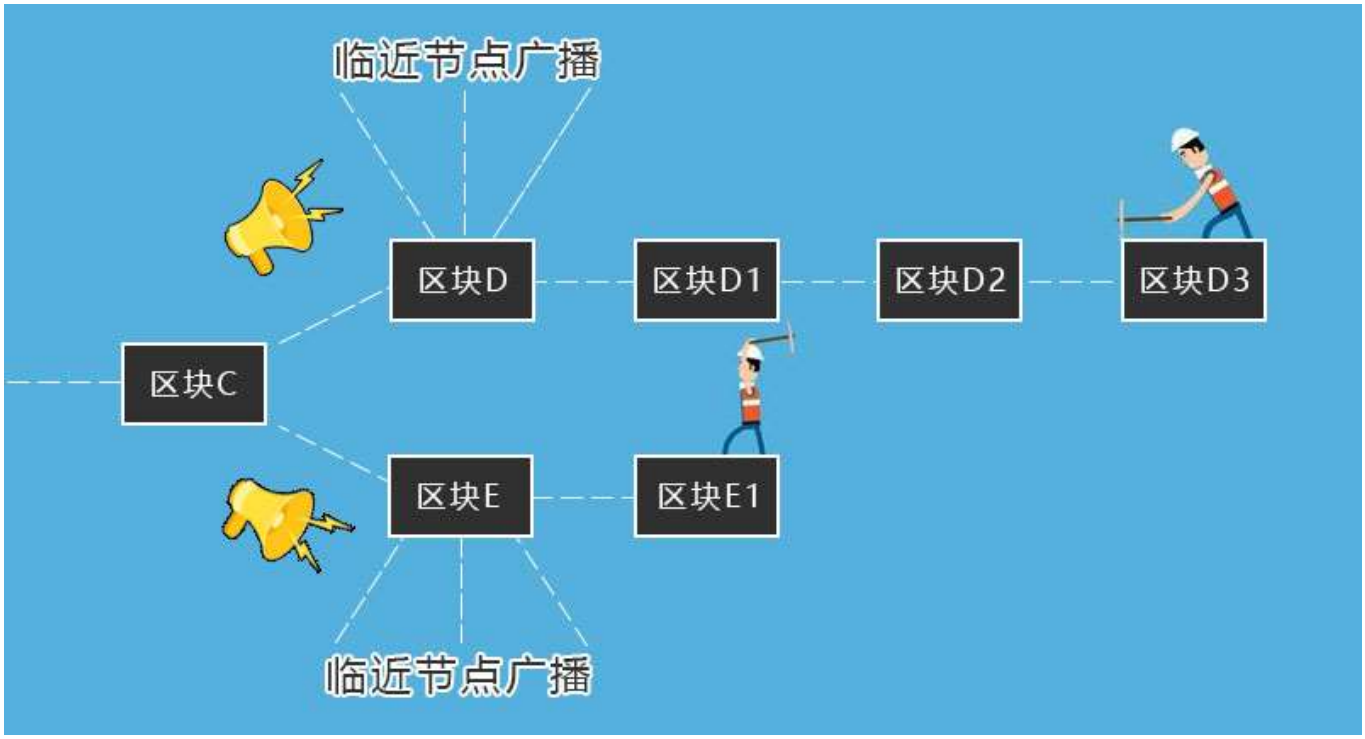
由于广播的行为，全网所有的节点都保持着相同的账本。

如果只有一个节点赢得记账权，全网广播后，大家都保持相同的账本，这不会出什么问题。

但是**广播是有延迟的**，假如有两个节点在几乎同一个时刻赢得了记账权，这两个节点都把交易数据打包进区块内，都向附近的节点进行广播，这就好玩了，**同一时刻产生了两个区块，**比如，区块D和区块E。



区块D和区块E会分别向自己临近的节点广播：“我是新生成的区块呀！快把我记录进去呀！”并且会分别在此基础上开始生成新的区块：



这个时候，全网的区块链账本**开始出现分歧**，有的是D的账本，有的是E的账本。

就好像树干进行分叉了一样，我们把这种同一个区块链网络产生了两个不同账本的情况叫做**分叉**。

那么，附近的节点就会先后收到D和E的广播，那么应该维护哪条链呢？

在比特币的算法里，节点会自动选择**最长的链**，也就是说，附近节点会自动维护D所在的区块链。

那么随着维护D的节点越来越多，维护E的节点越来越少，维护E的节点会发现，**这条链已经没有价值了**，继续挖下去只会白干。

于是最开始构建了区块E的节点，就会放弃这部分的区块，重新同步D的账本，E所在的这条分叉链就彻底作废了，区块链网络又达成了统一的账本。

在上例中，周围的节点都认可D而放弃E，E所在的分叉链存在了一段时间又消失。这种可以消除的临时分叉，就叫做“**软分叉**”。

所以，一般认为，比特币上的交易经过6个区块确认后就很难被颠覆了，挖到一个新区块别高兴太早，等后面连了6个其他区块的时候再庆祝吧。

比特币平均10分钟生成一个区块，“六次确认”大概需要经历1个小时。

二、区块的产生和连接

有些人头铁，不按照规则来，如果明知道D区块的链是最长的，偏偏要号召其他节点维护E区块的链，那么这个时候就形成“**硬分叉**”了，硬分叉是永久的。

除了这种因为系统自身原因不可避免的“自然分叉”，还有一种分叉叫做“**升级分叉**”。

要解释升级分叉的概念，我们必须要学习一下，比特币的区块是如何产生和连接的。

比特币有个叫**取哈希值**的算法，输入任何字符经过算法运算后都会输出一个**固定长度**的结果，这个结果就叫**哈希值**。

那么区块是如何产生的呢？

其实就是节点不停的进行哈希运算，输入这段时间的交易数据组成的字符和一个**未知数**，输出的哈希值一旦**满足某种规范**，那么这个未知数就被解出来了。

什么意思呢？我们初中的时候学过解方程。

有一个未知数，知道结果的情况下，去算出这个未知数就叫解方程了。
比如 x + 1 = 3，大家当然能解出 x = 2。

但是在哈希算法里，**知道哈希值，却无法反推出这个未知数。**

所以对于节点来说，让它求解 x + 1 = 3，它必须要让x从0开始一个一个的去试验，等式成立后才算解出来。

于是节点才会不停的一个数一个数的去试验，解出这个未知数，就赢得了第一个记账的权利，同时可以获得比特币的奖励。

然后节点就会把交易数据和这个未知数**打包成一个区块**，每个区块都记录上一个区块的哈希值，这样就达到了连接区块的效果。

可以发现哈希算法**验证未知数容易，计算未知数很难。**

三、升级分叉

如果每个节点都是算同一道数学题，比如要求哈希值前60位必须全为0，那么**新出的区块，所有节点都是承认的。**

可是区块链也是需要修改程序升级的，比如更新了一个程序把数学题改了，也就是说，本来需要哈希值前60位全为0，现在改为前100位全为0（难度比以前更大了）。

就好像由 x + 1 = 3，改为了 x + 1 = 5。

这就有意思了，由于**节点升级程序不是强制性的**，有的节点升级了程序，有的没有，这就导致**不同节点算的数学题不一样！**

这样就导致，计算 x + 1 = 5的节点产生的区块，不被计算 x + 1 =3 的节点所认可（因为一个解出x = 4，一个解出x = 2）。

于是，计算 x + 1 = 3的节点依旧在累加区块，计算 x + 1 = 5的节点也在累加区块但不被计算 x + 1 =3的节点认可，于是就**硬分叉**了，形成了两条区块链。

我们把这种由于节点升级程序不统一，导致的分叉叫做**升级分叉**。

对于互联网，只有一个节点，一台服务器升级程序就成功了。
对于区块链，有无数个节点，要想让无数个节点都达成共识升级程序，这就是一件非常困难的事情。

矿工都是有自己想法的，**升不升级，要看对自己有没有利**。假如升级的这个程序，算数学题难度变大了，需要更大难度才能挖到矿，那么部分矿工就不会响应开发者的号召，依旧挖不升级的矿，这样开发者往往会被矿工牵着鼻子走，但是却又无可奈何。

开发者和矿工达不成共识而硬分叉的典型案例就是 BTC 和 BCH。

四、自我升级

波卡又是如何解决节点升级的难题的呢？

林嘉文同志又一次看出了问题的根本所在，站在问题的本质因素去思考问题，这是林嘉文的伟大之处。

区块链升级之所以困难，之所以导致硬分叉，就是因为**各个节点不统一升级。**

因此只需要设计一套程序让节点每次升级程序时，没得选择必须强制执行，这样节点的升级才会统一，**再也不会出现硬分叉的情况**，这就是波卡的**自我升级。**

林嘉文把这套**自我升级**程序作为底层，包装在了Substrate框架之中。鸭哥之前说过，Substrate就像一个造链工厂，就连波卡也是Substrate一键发链造出来的，因此不光波卡，但凡是基于Substrate研发的**波卡的生态项目**，**波卡的平行链**也都具有自我升级的特性，**都永远不会硬分叉！**

有了自我升级的特性，波卡使项目能够保持高度灵活性，能随着技术的发展而发展，大大降低了有争议的硬分叉带来的风险。

——END

鸭哥创办了Polkadot新纪元社区，后续有千元红包和抽奖送DOT的活动，有一手消息或争取到的波卡生态项目的糖果也会作为福利发给社区，扫描二维码马上加入：

拉波卡群请加鸭哥V: cui1kcan2