Smart Bitcoin Cash：兼容EVM和Web3 API的比特币现金侧链

概述

比特币现金致力于提供去中心化、高通量、低成本、且易用的加密货币基础设施，对其主网的任何修改，都需要达到很高的共识，这阻碍了它进行试错、创新的进程。

因此，我们开发了Smart Bitcoin Cash——一条比特币现金的侧链，其目的是探索新的想法、解锁新的可能性。它将会同以太坊的EVM和Web3 API相兼容，因为它们已经是区块链DApp行业中的事实标准了。

以太坊通过逐步过渡到ETH2.0，正在解决其低通量、高成本的问题，但众所周知这个过程仍然需要若干年才能最终完成。Smart Bitcoin Cash尝试以一种不同的路径来解决这些问题：优化EVM和Web3的底层实现，以便充分利用硬件的能力，特别是其固有的高度并行的能力。我们相信Smart Bitcoin Cash能在更短的时间内实现ETH2.0的承诺：高通量、低成本。

动机

对于比特币现金，不同的人群期待不同的新特性。

比特币现金仍然坚持10分钟的区块间隔，相对于其他区块链的秒级出块，这显得太长了。尽管比特币现金支持安全的零确认交易，但是一些复杂的、支付以外的场景，仍然需要较短的确认时间才能达到好的用户体验，DeFi就是这样一种场景。

比特币现金有一个受限、非图灵完备的脚本系统。这使得它相对于以太坊的EVM更加难以使用。另外，毫无疑问，EVM和Solidity提供了最佳的生态系统并且拥有最多的开发者，足以傲视其他智能合约开发平台。比特币现金的生态中无法使用EVM是非常遗憾之事。

迄今为止，比特币现金已经证明其区块大小可以达到14MB。尽管现在它的区块大小基本在0.8MB左右，但增长的速度非常快，自2021年开始以来，已经增长了3倍。如果它继续保持这样的增长率，在不远的将来，它的区块大小就会稳定地超过14MB。由于大于14MB的区块仍然有很多未知问题，并没有在实践中检验过，最好现在就能提前为进一步的扩容进行一些准备和探索。

Smart Bitcoin Cash对比特币现金的贡献是什么呢？它为新的功能提供了试验场，它拥有更短的确认时间，它不但支持EVM和Web3，而且支持得比现有方案更好（通量更高）。它提供了一个新的渠道从而邀请更多的开发者和用户加入到比特币现金的生态系统当中来。

随着Smart Bitcoin Cash变得越来越成熟，为它所开发的库，以及从它身上学到的经验教训，也能帮助比特币现金主网的发展。

背景

Vitalik Buterin于2013年首次提出以太坊，迄今已有八年。八年间，智能合约诞生、发展，逐步繁荣。如今以太坊已是最为成功的智能合约平台。对它的生态系统进行调研之后，我们有如下的一些观察：

单链的用户体验，是Sharding和Layer2方案所难以匹敌的。零延迟和原子性的智能合约互操作，只有在单链的场景下才能实现。跨Shard或者Layer1和Layer1之间的交互必须经过链间通讯的过程，这一过程将引入类似于在中心化交易所充值和提币的延迟。一些著名的机制，例如flash loan 和flash swap，在跨链时无法工作。

低通量阻碍了普通用户同DApp直接交互。每个区块的Gas上限是固定的，矿池会优先打包那些Gas Price较高的交易。同时，低Gas Price的交易需要等待很长时间才会被打包，甚至永远都不会被打包。自然，只有那些高价值的交易才值得高Gas Price。因此普通用户在无法承受高Gas Price的情况下，只能将自己的资金存入中心化的机构，委托它们代替自己操作这些资金。在DeFi世界中，只有少数的账户在发送交易。这些账户拥有大笔资金，或者因为自己拥有很多，或者因为收集了委托者的资金。这是蛮讽刺的一件事，如今的去中心化金融并没有那么地去中心化。

相对于执行智能合约所进行的计算，存储消耗了更多的资源。在以太坊的历史上，存储操作的Gas已经被两个EIP增加过：EIP-1884和EIP-2200。但这仍然不够，有研究表明以太坊仍然低估了某些存储操作，使得它容易受到DoS攻击。因此，另一个EIP-2929即将在下一次的柏林硬分叉升级中，进一步升高存储操作的Gas。与此同时，EVM当中的256位长整数算术运算，在由Martin Holst Swende和Paweł Bylica所开发的库当中，得到了显著的提速。

链外的QPS（Query Per Second）同链上的TPS（Transaction Per Second）同等重要。一个DApp工作时，不仅要发送交易，而且要查询链的最新状态及其历史事件。交易只执行一次，但与之对应的事件和状态更新可能会被查询很多次。因此对QPS的总需求远高于TPS。在以太坊的生态中，查询操作通过Web3 API来完成。而Infura是Web3的最大提供商。Infura维护着一份Web3的优化实现，它比标准的全节点客户端（例如go-ethereum）更强，因此并没有开源。很多开发者都选择使用Infura的低成本服务，而不是自己运行全节点。这导致了，当Infura出现了严重的服务中断时，很多DApp和交易所就不能工作了。

用户对于交易延迟容忍度较高，同时并不十分关心交易的具体排序。通常，我们使用像MetaMask这样的钱包来签署交易，选择合适的Gas Price，然后广播交易。为了节省Gas费，我们经常会选择一个较低的Gas Price，然后等待数分钟甚至数小时直到交易最终被确认，而不是选择极高的Gas Price让它下个块就被确认。有时候其他人的交易会比你的交易更早确认，从而带来一些损失，例如更大的滑点。但大多数用户可以容忍。

我们还观察到自从2013年以来，计算机发展的最重要的趋势是：**CPU在多核的方向上走得更远了**。让我们来比较一下2013年和2021年的MacBook，同时预测一下2029年的MacBook。

	MacBook Pro in 2013	MacBook Pro in 2021	MacBook Pro in 2029
CPU核心数	2	8	32
CPU最高主频	2.9GHz	3.1GHz	3.3GHz
CPU制造工艺	22nm	5nm	1nm

集成电路工艺在28nm之后，已经很难继续大幅度提升频率，但是它的每一代升级，仍然提供了更高的晶体管密度。设计者利用这些晶体管，实现了越来越多的CPU核心。在过去的10年里，新创的编程语言都在宣传自己能够非常容易地开发多核CPU的潜力：Go语言的 channels and goroutines，Dart语言的isolates，以及Rust语言的fearless concurrency。

以上这些观察指导了Smart Bitcoin Cash的设计和实现。

Smart Bitcoin Cash的核心组件

Smart Bitcoin Cash的创新表现在若干程序库当中。它并没有发明很炫的共识算法或者密码学算法，而是采用了另外一种方法学：开发底层的程序库，以便充分地发挥硬件的潜力，尤其是其内在的并行性。普通的用户和开发者在一个支持EVM和Web3的兼容层之上操作，底层贴近硬件的优化被这个兼容层完全隐藏了起来。在编码实现中，我们使用了开发代号Moeing，它被附加到各个程序库的名称上作为前缀。

利用这些强大的程序库，Smart Bitcoin Cash中期的目标是将区块的Gas上限提升到十亿。长期的目标是通过Sharding和Rollup等技术进一步提升其通量。

MoeingEVM

MoeingEVM是一个并行执行引擎，它并发地管理多个EVM的上下文，同时执行多个交易。它的底层基于来自evmone的优化实现，同时还发明了若干新技术以便使交易的并发度最大化。

随着多核CPU变得越来越流行，以太坊风格的单线程执行引擎成为了可扩展性的障碍。如果我们切换到多线程的执行语义，那么多核CPU的开发将更加容易、方便和直接，最终获得更优的可扩展性。

因此，MoengEVM是按照多线程执行语义来开发的。

为了充分利用现代硬件中固有的并发度，我们尝试从两个方面来利用并行性：

1. 共识引擎和交易执行引擎可以同时运行
2. 不同的交易可以同时执行

为了让共识引擎和交易执行引擎同时运行，MoeingEVM采用了这样的策略：当一个新的区块被提交时，其内部的交易并没有被立即执行；它们只是被存储了起来，作为世界状态的一部分；之后世界状态的Merkle Root被计算出来让下一个区块可以被打包和投票，与此同时，之前被存储的交易会被检视和执行。

为了让多个EVM同时执行，我们运行来自不同区块的交易被混合和重排序。在每个轮次中，一组（bundle）不相关的交易被取出和并发执行，以最大化并发度。经过若干个轮次之后，所有的交易都被执行，或者残留一些交易尚未被执行，这些残留的、未被执行的交易被重新保存到世界状态中，等待之后被执行。由于交易被强制地划分为组，而且组内的交易都是并发执行的，这种策略被称为“**强制组并发**”。

MoeingADS

为什么存储操作如此昂贵呢？以太坊的存储引擎MPT是根本原因。

MPT是不可或缺的吗？未必，很多区块链（包括比特币现金）不使用它也能工作得很好。但是，作为一个可验证数据结构（Authenticated Data Structure），它可以证明状态的存在性和不存在性，这对于实现去信任化（Trustless）非常重要，轻客户端和跨链都以此为基石。

因此我们开发了MoeingADS，一个可以替代MPT的可验证数据结构。

以太坊的存储引擎采用了双层架构。第一层是LevelDB，第二层是MPT。而其他的一些区块链，例如比特币和比特币现金，使用单层的存储架构：它们直接使用LevelDB来存储UTXO结合。MPT在LevelDB提供的功能之上，实现可验证数据结构，其付出的代价是更低的读写通量。每次EVM读写世界状态，MPT需要对LevelDB执行若干次操作，而每个LevelDB的操作都需要访问若干次磁盘。这样整体的性能就降下来了。

MoeingADS使用单层架构，它直接访问底层的文件系统，不依赖于其他数据库。它是一个可以直接提供存在性和不存在性证明的键值数据库。使用MoeingADS，读取键值对只需要读一次磁盘，覆盖已有的键值对需要一次读和一次写，插入新的键值对需要两次读和两次写，删除键值对需要两次读和一次写。而且，这里所有的写操作都是在文件末尾追加，这对于磁盘而言非常友好。

实验表明，MoeingADS比LevelDB还要快。它为此付出的代价是更多的RAM消耗：每个键值对需要16个字节。

MoeingDB

除了用来支持MPT，LevelDB还被广泛地用于保存历史信息，例如区块、交易存根和日志。然而，LevelDB并不是为了区块链的工作负载而优化的。区块链的工作负载具有如下的特征：

1. 读操作的数量要远多于写操作
2. 没有必要支持Read-Modify-Write的原子交易
3. 简单的读写锁比复杂的MVCC更好，因为修改操作都是按区块进行的批量写
4. 由于大多数键都是Hash ID，空间局部性不好，导致很难实现有效的缓存
5. 容易被DDoS攻击，除非冷数据的读延迟具有合理的上限

MoeingDB是一个领域定制的数据库，专门用来存储区块链的历史，它专门针对上面的负载特征而开发。基于它的能力，可以开发出一个完全开源的、高QPS的Web3 API，这对于Smart Bitcoin Cash和以太坊都有正面意义。我们希望它可以使得Web3 API供应市场变得更加去中心化。

MoeingKV

MoeingKV是一个比LevelDB更快的键值数据库，为了提升速度，它只支持读写操作，不支持迭代。

为了支持迭代功能，LevelDB进行了很多折衷设计和专门的优化。但是在绝大多数情况下，区块链存储引擎并不是一定要依赖迭代功能，例如以太坊的MPT和比特币现金的UTXO存储。

在底层数据结构设计和代码实现过程中，MoeingKV为了加速普通的读写操作而进行折衷和优化。它可以替代LevelDB，作为支持MPT的底层库。

在某些场合下，由于兼容性的原因，MPT不能被MoeingADS所代替，这时可以使用MoeingKV来支持MPT。以MoeingKV为底层的MPT比MoeingADS要慢，但它比以LevelDB为底层的MPT还是快多了。

MoeingKV的一些关键思想来自于MoeingADS和MoeingDB，但它并没有被用于Smart Bitcoin Cash。我们开发它，主要是希望其他项目可以从中获益。

MoeingAOT

MoeingAOT是EVM的AOT（Ahead-Of-Time）编译器。

在区块链领域，EVM相对于所有其他的虚拟机（例如WebAssembly）都要更加流行，已经是智能合约事实上的标准了。

然而，EVM缺乏一些重要的加速方法，例如AOT编译器和JIT（just-in-time）编译器。在软件行业，几乎每一种重要的VM都有其AOT编译器和（或）JIT编译器，例如：JVM、ART VM、Javascript V8、DartVM、WebAssembly、LuaJIT和GraalVM。我们相信EVM足够重要，值得拥有它自己的编译器了。实现一个AOT编译器是更容易的选择，因为EVM同Javascript和Lua这些动态语言不同，它是静态的。

MoeingAOT将EVM字节码编译为机器码，这些机器码被保存为动态链接文件。在EVM解释器启动一个智能合约之前，如果它找到了对应的动态链接库，它会载入这一动态链接库执行机器码，解释执行就不再必要了。

对于常用的合约，例如USDT和UniSwap，AOT编译器是非常有用的，因为机器码比解释执行快很多，它能够大大缩短合约的执行时间。

MoeingRollup

Rollup是一种"去负载"（offload）的方法学，用来提升区块链的通量。不同的项目（例如 optimistic rollup和arbitrum rollup）有不同的实现。一般而言，Rollup意味着将整个状态集合归结为一个哈希值，即状态根。通常，一个Rollup扩展实现在智能合约内部，由一个定序人（sequencer）来维护其状态，他要负责把用户的交易打包，提交每个区块的状态根到合约中。相邻区块的状态根必须符合状态转换的规则，第三方能利用一些证明数据来验证这一点。

一个诚实的定序人必须可靠地维护状态，中立地打包交易（不能进行审查）。他还必须向所有需要下载状态和区块的人提供数据。否则，定序人就被用户驱逐，由新的定序人替代，这一过程借助智能合约中定义的Staking机制来实现。

进一步，如果相邻区块的状态根不符合状态转换的规则，会有人向定序人提出挑战，定序人则必须给出证明数据。如果他做不到，将被罚款和逐出。

智能合约中定义了一个Rullup扩展的运行规则的细节，这些细节千差万别。但不论如何，它们核心的机制是通过证明数据来确认状态的转换是合法的。不幸的是，这一证明任务对于EVM而言太重了，实现起来很困难。

MoeingRollup使用原生代码来实现这种证明任务。智能合约中利用它所提供的原语，可以简单且高效地完成证明任务。证明数据中，至少包含如下三个部分：

1. 区块中的交易集合
2. 这些交易读取的键值对，以及它们的存在性证明
3. 这些交易输出的键值对，以及一些辅助数据，可以利用它们来生成新状态的状态根

MoeingRollup还提供了一些工具用来方便定序人的工作，例如生成证明数据等。

MoeingLink

Smart Bitcoin Cash以单Shard的方式启动，但长期来看，它在未来有可能包含更多的Shard，从而成为一条多Shard的区块链。

MoeingLink是一个协议，它允许多个Shard可以直接互操作，而不必在比特币现金的主网上执行交易。

当前，所有主要的多Shard方案都需要一个链做中转。在ETH2.0方案中，由Beacon Chain来做；在Polkadot方案中，由Relay Chain来做；在Cosmos方案中，由Hub Chain来做。随着越来越多的Shard被创建，跨Shard的交易会给拥挤的Layer1造成很大的压力。

为了避免这些情况，MoeingLink允许各个Shard向其它Shard证明自己的状态，方法是将MoeingADS中生成的世界状态的Merkle Root提交到Layer1之上。各个Shard能获知彼此的状态之后，它们即可在没有Layer1协助的情况下进行交互。

共识算法

Smart Bitcoin Cash使用tendermint作为共识引擎。矿工和BCH持币人共同来选举验证者集合。在时间上，验证者集合按照epoch来进行选举和履行职责。

一个epoch持续2016个块（大约两周时间）。在一个epoch内，BCH的持币人通过施加时间锁的UTXO来证明自己的拥有权，并且使用其拥有的数额来为验证者投票；而矿池则使用coinbase交易来投票。这是一种混合的共识模型：算力和币混合投票。投票的过程在比特币现金的主链上进行，并且是无许可的：成为新的验证者只需要具备足够算力和（或）拥有足够的BCH。

一个epoch的结束时间是其中区块的最大时间戳，它的持续时间是相邻两个epoch的结束时间之差。在一个epoch期间被选举出的验证者集合，会在一个"候任状态"中保持一段时间，其长度等于epoch的持续时间的5%。之后，它们才会开始履行职责，等到下一个验证者集合的"候任状态"结束之时，它们停止履行职责。

每个验证者都必须在Smart Bitcoin Cash侧链上锁定一些BCH作为抵押物。如果它在履职期间有不当的行为，这些抵押物将被罚没。

Smart Bitcoin Cash启动后的第一个阶段，只有算力可以用来选举验证者集合。在主网上锁定BCH来选举验证者的功能会在之后开发，并且通过一次硬分叉升级来生效。

代币和燃料

Smart Bitcoin Cash不会引入新的代币，它的原生代币就是BCH，而且使用BCH来支付Gas费。

交易的Gas费被收集之后，其中一半会在验证人集合卸任之时奖励给他们，另外一半则会被燃烧掉。这一设计使得BCH成为一种通缩的货币。验证人必须有足够的BCH作为抵押才能获得Gas费奖励，这些奖励在可以被验证人花掉之前，要经历一段锁定期。

回购后燃烧的机制，在平台币（BNB、HT、FTT、OKB等）和DeFi治理代币中非常普遍。Filecoin使用了类似的机制来燃烧掉部分Gas fee，以太坊在硬分叉实施EIP-1559之后，也会实施这样的机制。这一机制已经被证明是有效的，所以Smart Bitcoin Cash也会采用。

BCH可以在比特币现金主网和Smart Bitcoin Cash之间双向互转。我们可以在主网上锁定若干BCH，同时在Smart Bitcoin Cash上解锁对应数额的BCH，反之亦然。为了能启动Smart Bitcoin Cash，我们正在邀请比特币现金生态中的重要参与者一同来运行一个联盟式的双向楔入的网关。这个网关连接主网和Smart Bitcoin Cash，BCH通过它双向流通，类似于RSK和Liquid所采用的机制。网关的参与者不一定要在验证者集合中。

如今的加密社区已经非常适应"同一代币存在于多条链"的范式。例如，当提到USDT的时候，它可以指比特币的Omni协议上的代币、比特币现金SLP协议上的代币、以太坊上的代币，或者波场上的代币，等等。因此，我们不会使用另外的符号来表示Smart Bitcoin Cash上的BCH，以免引发误解。

我们明确知道BCH的脚本语言足以实现非托管的、去信任的网关，通过由一段锁定脚本来追踪在coinbase交易当中进行的投票进程。然而这一技术路线并未在实践中被验证。我们将会撰写专门的文档来描述这一技术，当它通过社区的评审之后，再使用CashScript来实现它。接下来，Smart Bitcoin Cash将会通过一次硬分叉来切换网关。

同比特币现金上Layer-2方案的互操作

再比特币现金生态中，已经有若干中layer2扩展，支持发行同质化和异质化的代币。在这些扩展中，最为成功和重要的是Simple Ledger Protocol。发行者在代币的生态中起到最为关键的作用，他可以帮助进行代币在SLP和Smart Bitcoin Cash间的互转。

例如，如果Alice希望把10个XYZ代币从SLP转移到Smart Bitcoin Cash，她可以先将这些代币在主网上发送给XYZ的发行者，然后发行者再在Smart Bitcoin Cash上把10个XYZ发送给她，反之亦然。为了让这个过程更加安全，Alice可以使用原子交换来保证两处的代币转移要么都发生，要么都不发生。

路线图

MoeingADS、MoeingEVM和MoeingDB已经基本开发完毕，在Smart Bitcoin Cash正式启动之前，它们还要经过详尽的测试。

MoeingAOT会在2021年底之前开发完毕，并且通过一次硬分叉升级来生效。MoeingKV同样会在2021年开发，希望它能为比特币现金主网的扩容贡献一些力量。

MoeingRollup和MoeingLink将会在2022年开发。如果那时Smart Bitcoin Cash出现了拥堵的情况，它们会被实施，以实现进一步的扩容。

结论

Smart Bitcoin Cash是一条兼容EVM和Web3的比特币现金的侧链。比特币现金的算力和持币人为它选举验证者集合，它使用BCH作为燃料。它使用若干对硬件友好的程序库来实现扩容。我们相信它能够在更短的时间内提供和ETH2.0相同的好处，使得单区块的Gas上限达到十亿。

我们开发了激进的技术来优化存储和执行引擎。Smart Bitcoin Cash，在很大程度上可以被视为这些新的激进技术的实验场。同其他开源项目一样，它的设计和实现可能会有缺陷和漏洞。因此，当您将资产（包括BCH）转移到Smart Bitcoin Cash侧链上之时，需要自行承担风险，保证您能够承受可能的损失。