SmartX 技术白皮书

1. 项目介绍

1.1. 关于 SmartX

SAT（SmartX 智图）是由 SmartX 社区团队研发的区块链技术公链，是一个去中心化、支持 Pow 挖矿、“基于交易 DAG”结构的 BlockDAG 项目，可实现性能无限扩展，本质上等同于平行扩展的传统区块链系统，可以无限平行扩展。

基于 SmartX 平台可无缝将区块链技术直接应用于加密社交、私密通讯、游戏支付、卡券红包积分等产品，通过区块链网络链接全球服务商与用户，以社交娱乐为切入口构建基于可信任并且安全的社交娱乐生态。

未来的 SmartX 平台是一个资金流、信息流、价值流多重平台，在 SmartX 构建的信任价值体系中，各式各样的人或物将自我价值通过区块链网络进行传递，形成丰富的价值互联网络生态，这最终将极大地提升社会生产效率。

1.2. 项目概览

区块链系统发展到如今，交易速度成为了区块链产业化如 DAPP、物链网、可信任数据交换等进一步发展的瓶颈。BTC/BCH 等纯粹通过增大区块大小而达到扩展目标，此方法手段不可取。另外通过 BTC/ETH 子网络的方式如闪电网络，跨链等技术其实是在牺牲安全性的前提下提高 TPS，同时必须具备更多的附带条件（如交易双方必须同时在线，需要抵押等）才能交易。那么就产生了另一种技术解决方案就是 BlockDAG 方案。从最早的 IOTA 到近期的 Conflux 项目，还有就是本团队的 SmartX 项目。

SmartX 有一个自己的原创性，具备创新性且全新的 DAG 结构性算法---分区出块懒融合算法（B-DAG 算法），可以把全世界节点产生交易分区融合起来，而跟目前主流 BTC/BCH，Ethereum 等主流 PoW 币区块必须同一时间 Pending 住交易然后挨个打包，不同节点打包交易区块互斥（虽然 Ethereum 有 GHOST 和 Uncle Block 机制，但远远不够）所有不同。

相当于高速公路只有一条出口，不管道路有多少条，行驶车辆有多少，最终只有一条出口。而出口速度决定了车流量，后面只能排队。SmartX 相当于开辟了无限个高速公路让车流量可以无限扩展。

通过 Epoch 周期产生一个稳定主单元 MC，按照 B-DAG 算法线性链接在一起，从主单元角度形成类似比特币链式结构。但整个视图是交易 DAG 结构，可以分成独立 Partition，互不干扰产生交易而后进行交易块融合，能够无限扩展性能。

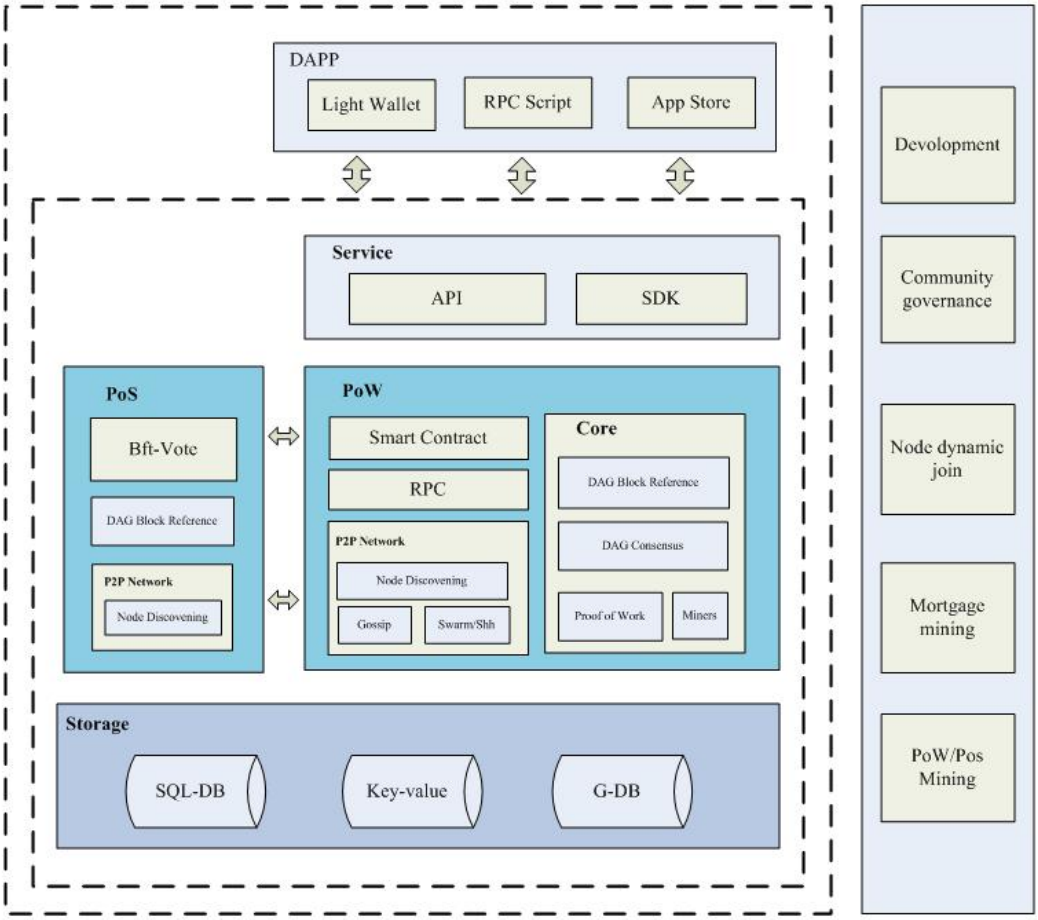
在这个结构中，双花检测迟早能成功，在全局角度算法是稳定一致的。一旦一个节点可以看到某个 Epoch 的全局数据，加上一个稳定一致排序算法（B-DAG 算法）那么双花检测就会成功。

1.3. 系统结构

中心化系统好比集中式教堂一样，必存在一单点故障，如果单点崩溃则系统崩溃。区块链系统则类集市，多个中心，人员嘈杂且有统一之市场规则。完全去中心化则意味着没有Owner，没有中心化利益方。优点避免中心化带来之单点作恶和单点崩溃现象。缺点是付出极大能源消耗和资源浪费。

若做一笔交易，在中心化云服务器如微信支付，不到一秒时间，消耗磁盘和网络带宽资源不过 1k 字节，能源消耗可以忽略不计。但是在 ETH 系统中，按 1 万个节点计，其硬件资源消耗则相应是 1 万倍，能源消耗可能百万千万倍。

在 SmartX 中，我们采用教堂和集市部分优点。系统架构按联邦模式运作。主网络由多个质押代币的 PoW 主节点和多个质押代币的 PoS 裁决投票节点组成。PoS 节点由社区有声望之组织竞选加入。PoW 主节点通过抵押一定量的代币自由加入和退出。PoW 主节点相当于矿池节点，可以接受矿机各种算力提交，交易，账本下载等服务。外围是普通节点，用户可以加入普通节点或者使用轻钱包，RPC 脚本，SDK API，或者 APP Store 上的应用接入到主网中使用 SmartX 服务。



1.4. 交易验证和确认

一个交易 TX 首先会发往距离它最近的一个主节点 NodeA 的内存池中，且在某个 T 周

期中，NodeA 会产生一个主单元 M 主动去引用交易 TX，同时会把主单元 M 和交易单元 TX 组成的一个大块 M-muster，广播到所有主节点。其他主节点同时也会做类似的操作，最后只有一个工作量证明最大的 M 主单元，成为胜出主块 MC。跟 BTC/ETH 等 BlockChain 技术不同，其他低工作量证明的主单元 M 不会失效，他们将会被胜出主单元 MC 直接或者间接引用，形成了一个 B-DAG 的树图结构。

当 T 周期的 MC 块确定时，所有被 MC 块直接或者间接引用的 M 块和 TX 块则被确认。在选 MC 的过程中是一个典型的 PoW 过程。

PoS 共识则是一个 3F+1 的一个选举人网络，难度最大的 M 单元在成为胜出 MC 时，须得到 PoS 选举人网络的裁决确认。B-DAG 的树图结构因为有了 PoW 的链上共识，可独立运行，SmartX 在 PoW 共识下运行，跟 IOTA 等 DAG 项目没什么不同。

为了加速 B-DAG 结构的收敛（如多网络下的分叉融合），SmartX 引入 PoW/PoS 双层共识，PoS 层相当于一个网络加速器。对 PoW 共识做加速确认。

PoS 投票层的引入不仅可以对 B-DAG 动态结构进行收敛，同时也可对其他链上共识进行投票如余额快照，做数据裁剪，不可逆节点，时间共识，节点 QoS 服务等。为了奖励 PoS 节点对主网的服务，每产出一个奖励块时，PoS 也会分到固定的奖励。

采用 PoW/PoS 联合共识优点在于，能让 DAG 网络在保证去中心化，不丧失安全性和高 TPS 前提下，处理实际业务问题。

1.5. 块图技术和区块动态扩容

由于 SmartX 主节点并发出块，每个主节点 MC 会主动引用所有本节点所产生之交易，理论上各个主节点都可以独立出块（MC-Muster），该块为单 MC 引用之所有 MC 和 TX 的集合 MC-Muster，受限于和计算机内存和网络宽带，理论上 MC-Muster 伸缩性极强。小至一笔交易，大至几千几万笔交易皆可打包至一个 MC-Muster 里。因此 B-DAG 技术方案亦可作为传统 BTC 链式结构区块链的一种动态扩容方案之一。

高 TPS 已然成为区块链发展之最大瓶颈。类似 Fomo3D 游戏，或高频链上抢红包游戏皆需要高频 TPS，SmartX 尝试通过 B-DAG 技术解决此类问题。

2. SmartX 的 DAG 结构

2.1. DAG 结构

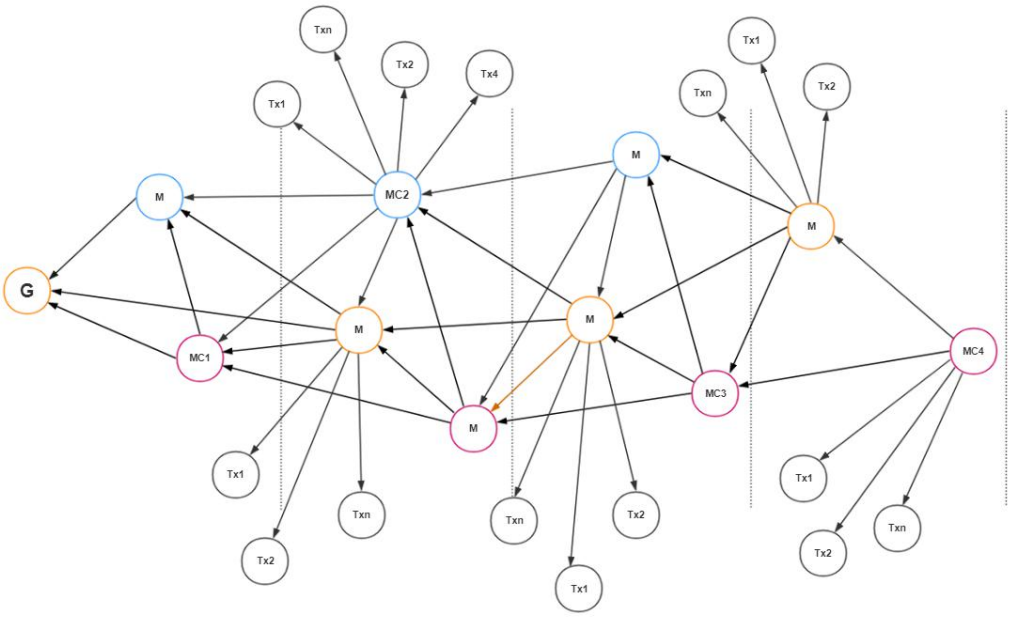
与传统 BTC 链式区块链不同，SmartX 区块 Block 即交易 TX，交易 TX 即区块 Block。除 TX 交易单元外，SmartX 系统还存在 MC 主单元，MC 主单元根据一定周期 Epoch 随机产生用于引用之前产生的所未引用有交易。虽然直观看，MC 和 TX 组成 DAG 结构，单从每个 Epoch 看，随着时间流逝，如果只保留 SAT 主链引用（后面有介绍），其他 MC 引用线去掉，则接近于打散了交易的 BTC 链式结构。该结构结合 BlockChain 和 DAG 技术部分优点。本文

称之为 B-DAG 技术。

跟大部分区块链系统类似，通过手续费来杜绝 DDOS 攻击。由于 B-DAG 技术模型不需要全网 Pending 打包一大区块，即无法控制交易量，交易量到达巨高点而引发雪崩。解决此类问题方法即把控制交易量阀门交给主节点手里。主节点根据自身节点性能指标去动态设置阀门而从达到动态扩容的目标。

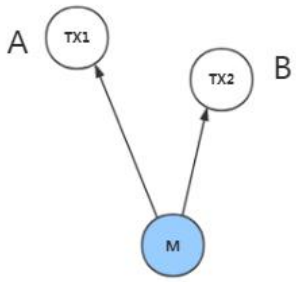
主节点：即网络出块打包的挖矿节点。此节点跟 BTC/ETH 矿池节点类似，但必须通过抵押一定数量的 SAT 币即可加入网络。

一个典型的 B-DAG 结构如下：



2.2. 简单区块结构

主块 M 引用了两笔交易 TX1 和 TX2，并且经过 PoS 节点 A，B，C，D 签名确认。



如果我们使用 Json 格式来描述一个简单主块引用且经过了 PoS 节点签名的结构则：

```
{
    "hash": "THIS HASH",
```

```
    "time": "2019-08-06 19:10:18",
    "type": "1",
    "diff": "THIS diff",
    "owner": "OWNER ADR OR PUBKEY",
    "nonce": "5CC911F3B8434911B691B6CF7A361333",
        "flds": [{
        "type": "SAT_FIELD_OUT",
        "hash": "A",
        "time": "2019-08-06 21:21:19"
    }, {
        "type": "SAT_FIELD_OUT",
        "hash": "B",
        "time": "2019-08-06 21:21:19"
    }],
    "Signers": ["A", "B", "C", "D"],
    "Signinfo": "SIGN CONTENT",
    "sign":"THIS ECDSA SIGN"
}
```


2.3. 账户模型

跟传统交易系统一样，SmartX 使用余额账户模型。每一个 Epoch 共识周期处理发现某个 TXBlock 之 OUTPUT 账户不存在，则全网创建此账户。并将 INPUT 的 Amount 转入此账户中，而 INPUT 需存在且余额足够。

每个账户余额由交易单元 INPUT 和 OUTPUT 之差决定。在某个 Epoch 中，从创世块至该时刻，账户余额 Balance = All(INPUT) – All(OUTPUT。每个交易单元由 INPUT 方用 ECDSA 私钥签名，用 INPUT 公钥进行验证块合法性。

系统保证每笔 Transcation（后面称 TX）经状态机处理为幂等性，即交易 TX 执行一次和执行多次最后结果都为一样。修改余额时需并发加锁，系统每个过程和函数都需具备可重入性。于是乎，账户余额修改跟状态机一样，不管流入多少笔交易，又或者多少次重复交易，其最终结果都为一致。

判断是否为重复交易即根据交易 TX 之随机数 Nonce 值，除了 Nonce 一样，整包 hash 值一样称之为重入成功，反之称为重入失败，系统对这两种情况分别处理。

同时账户表中，每个账户会有一个 Nonce 字段，用于检测每个转出交易的重复交易，但如果重复交易在不同池子中，经过稳定排序，优先高的块先执行，优先低的块不存储实际块，只存储哈希。

后续更新块时，先更新优先级高的块，只有哈希没有实际块的交易，之前一定会存在重复交易，要检查这种情况。

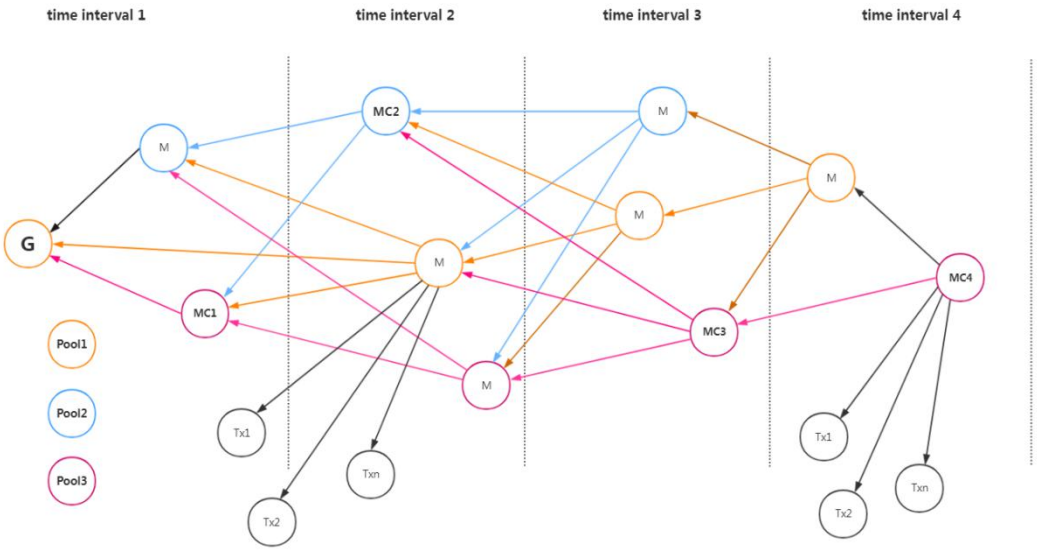
2.4. 引用关系

全网主节点按一定的 Epoch 并发产生主单元 MC，及该 Epoch 下所产生所有交易单元 TX，所有 MC 和 TX 按约定规则进行互相引用，总规则如下：

- 约定规则 1：主节点自身产生主单元可以引用其他广播过来主单元，但广播主单元不能引用目标主单元。
- 约定规则 2：主单元可以引用交易单元，交易单元被某个主单元引用后不能再被其他主单元所引用。
- 约定规则 3：主单元之间不能循环引用。

自身主块：在某个主节点中按一定 Epoch 产生之主块，并且主块主动搜索内存池中所有未被引用广播主块和交易块。

广播主块：目标主单元产生并且完成引用操作后，广播给连接节点后该主单元变成广播主单元。



如图所示，SmartX 采用向前引用的规则，通过周期来区分是否引用其他矿池广播过来的主块：

- 如果广播主块和当前主块处于同一周期或者当前主块周期之后，则暂时不引用，而是等到下个周期再引用广播主块
- 如果广播过来的主块在当前主节点之前，如符合引用条件，则引用该主块
- 所有 M 块（包括胜出 MC 块）必须引用上一次 Epoch 周期裁决 MC 块
- 上一个 Epoch 周期不被 MC 块直接或者间接引用的主块会会被主网抛弃

3. SmartX 共识

3.1. PoW-PoS 双层共识

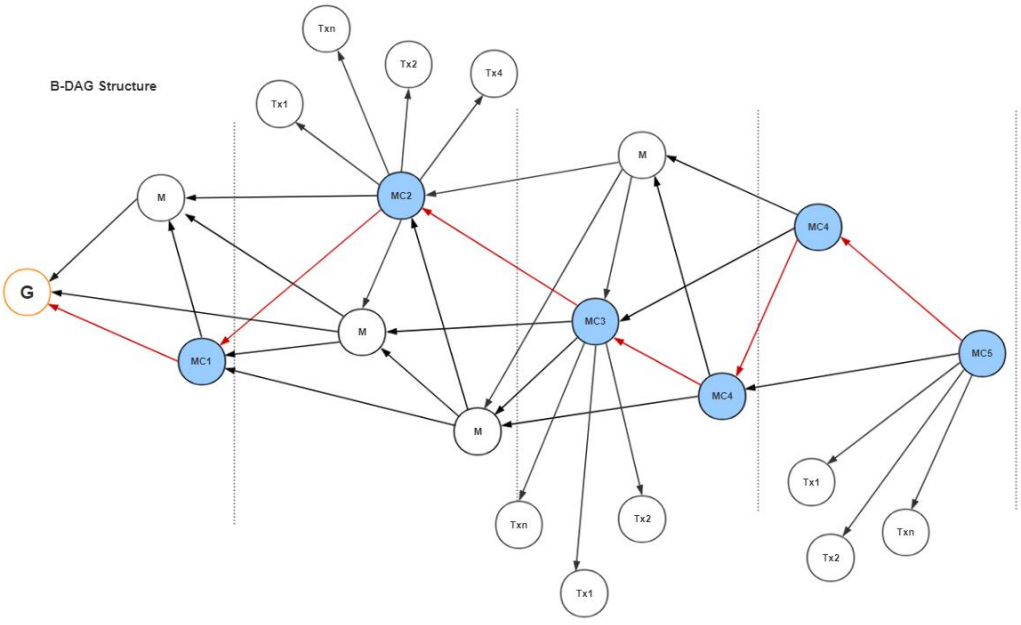
在 SmartX 中，采用的是 PoW 联合 PoS 的双层共识。PoW 的链上共识，每个时间片决胜一个 MC 块即工作量证明难度最大的主块为胜出主块。PoS 层是一个 3F+1 的 BFT 共识网络。难度最大的 MC 块同时也是 PoS 选举人投票出来的胜出主块，PoS 层类似一个加速器，可以加速交易的确认。

PoS 层除此之外还可处理不可逆节点，全网余额快照，PoS 服务，惩罚交易，时间校对等各种类型的全网共识投票。在孤立网络分裂时，主网有 PoW/PoS 双层保护。

3.2. 最大权重链

在 SmartX 中，每过一定 Epoch 周期产生一堆 MC 块，MC 块之间互相引用。由于可以通过 PoS/PoW 协议在一堆 MC 块中挑选出一个最大权重的 MC 块（正常来说都是 PoW 工作量最大的主块），称之为胜出 MC（主 MC），主 MC 根据 Epoch 周期在整个 DAG 结构中连成一条特殊的线，该线称为 MC 主链。

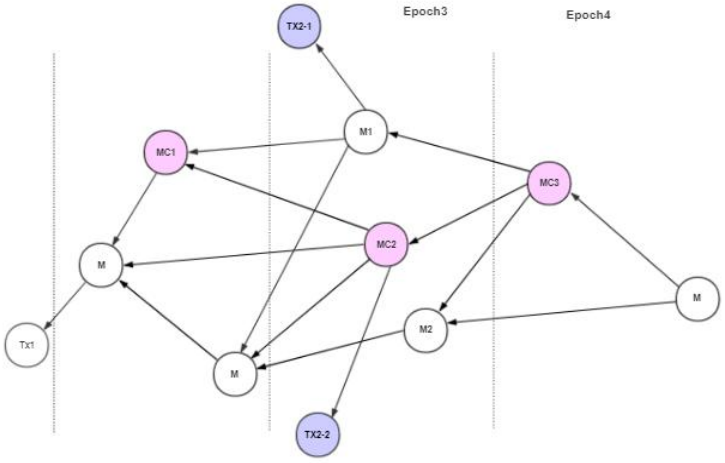
根据 SAT 主链可以排出所有交易 TX 序号（根据拓扑关系，时间和哈希值关系），根据此序号排除双花。



3.3. 双花排除

在图中，Epoch3 出现了两笔双花交易 TX2-1 和 TX2-2，由于 SAT 主链已确定 MC1<-MC2<-MC3，因此通过 MC3 通过拓扑排序可以判断 M1 序号优于 M2，从而 TX2-1 序号优于 TX2-2，到此 TX2-2 交易被排除，双花检测成功，双花检测排序规则如下：

- 能被 MC 块直接可达块，可通过块跟主 MC 拓扑序来排序
- 如果跟 MC 块没有通路，则通过最短路径排序即达到创世区块最近之路径
- 如上无法判断，通过块 hash 和时间来排序



3.4. Merkle 哈希

SmartX 每一个 MC 之当前 merkle_hash = hash(MC_hash + Pre-merkle_hash)，即自身哈希加上 MC 引用所有 Block 的哈希再哈希结果。跟 BTC 等链式区块链不同，SmartX 不存在一个单向可回溯 merkle 哈希。但每一个 Epoch 周期回头看 Epoch-N 周期时，每个 MC 的 merkle 哈希都必须保持一致，SAT 据此哈希来保证所有节点某一刻数据最终一致性。

3.5. 检查点和数据裁剪

随着时间流逝，区块链系统数据会膨胀到普通节点无法承担之重，鉴于此，理论上只要在某个检查点之前数据做一个快照，只要保留账户余额表和部分 Merkle 哈希用于校验。系统仍可以参与挖矿和交易，对于普通节点而言，这将大大减轻存储压力。

4. SmartX 市场和挖矿

4.1. 抵押挖矿

SmartX 谨慎设计经济模型。谨慎设计 PoW 挖矿和 PoS 挖矿奖励，以期在 SmartX 市值和 PoW/PoS 市场抛压中取得一个平衡。

SmartX 的默认初始化的几个 PoW 创世节点和默认抵押币 100 万个（PoS 抵押数更多），为了让运行意愿更强的参与者来参与到 SmartX 的节点运行，SmartX 支持挖矿节点的随时加入和随时退出，这样做的目的是为了更好的筛选出参与意愿更强的节点。

SmartX 的抵押挖矿会经过如下几个阶段：

- **初始化阶段**

  默认初始化几个创世主节点和默认抵押币 100 万个。检测在抵押表中是否有抵押标记，若无抵押则拒绝接收该主块

- **新节点加入**

  发起一笔抵押交易，内容：抵押币数，公钥，私钥签名，时间。经过大部分节点接收该块时，节点把信息写入到抵押表中。经过 2 分钟后（经验值），该普通节点可以参与挖矿。如抵押交易没有被某个主节点接收到则该节点产生之主块会被拒绝。

- **节点退出**

  发起一笔反抵押交易，经过大部分主节点接收后，则抵押表抵押标记会被置空，表示该抵押币已被取出。该节点可以如期把抵押币转走，如果其他主节点发现抵押标记已经被清空或者不存在，则允许该节点余额被转走。

4.2. 智能合约

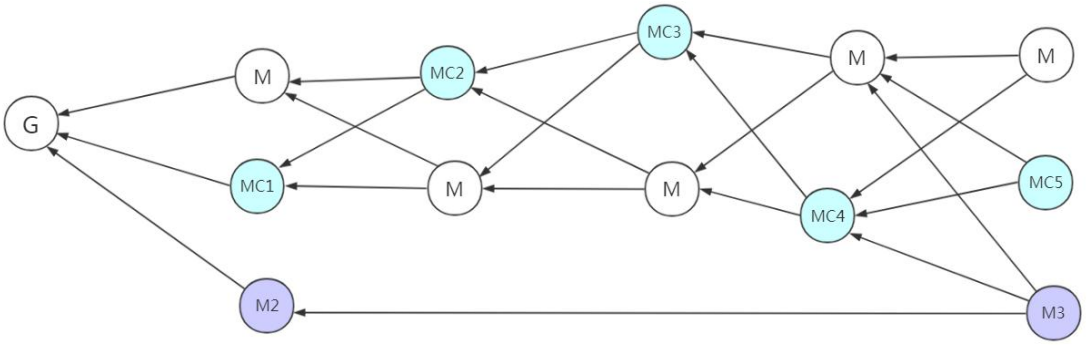
智能合约开发的 DApp 极大丰富了公链的生态环境，SmartX 将支持智能合约，暂时使用 EVM 虚拟机，支持智能合约开发工作会放到测试网上线后期再开发。

5. SmartX 安全性

5.1. 分叉链攻击和解决

存在两条分叉链，主链，G<-MC1<-MC2<-MC3<-MC4<-MC5 和私挖分叉链，G<-M2<-M3，其中 M3 的难度比 MC5 的累积难度高，在普通的 DAG 网络中 M3 会生成胜出 MC，MC5 这条主链将会被主网回滚，M3 链会成为真正的主链。BTC 同样可能出现类似问题，这就是 51%攻击，一旦某个节点算力成为大算力，那么则存在逆转交易的可能。

在 SmartX 这里，一旦检测到这种这种情况出现，SmartX 裁决层 3F+1 节点将介入处理。即不会让 M3 胜出。



5.2. 时间戳

跟大多数区块链系统一样，SmartX 系统有一个默认提前，大多数主节点时间戳基本一致。如果主节点时间跟 NTP 时间相差甚远，挖矿线程会被停止，且某主节点收到来自临近节点块时间超过 NTP 时间一定时间，块会被拒绝。如果节点时间误差在可容忍范围之内，被认为是网络异常导致延迟。

时间切片按“格林威治”时间戳每固定 T 周期做一个切片，所有根据当前时间戳产生主块映射到格林时间的时间切片中，如果都一致，则认为大家时间戳一致。

BTC 式区块链可能存在时间戳攻击缺陷。即下一个难度调整值等过去若干个块总难度除以头尾时间之差。由于时间戳不可控，攻击者可以通过伪造时间戳造成时间戳波动过大，导致难度调整巨幅波动，直接后果即网络无法出块有或是出块过快，变相增产。

SmartX 设计会避免此类情况，即周期律动出块，不调整难度。随着时间流逝最新 MC 难度等于创世快以来难度之和，通过比较总难度来确定主 MC。

5.3. 女巫攻击

女巫攻击（Sybil Attack）是 2002 年由 John R. Douceur 提出的，它是作用于对等（Peer-to-Peer,简称 P2P）网络中的一种攻击形式:攻击者利用单个节点来伪造多个身份存在于 P2P 网络中，从而达到削弱网络的冗余性，降低网络健壮性，监视或干扰网络正常活动等目的。

在 SmartX 网络中，为了解决来自恶意节点或者节点失效带来的安全威胁，每个消息都会要求消息发送者使用自己的私钥对消息进行签名，其中验证签名的参数包括消息内容 message，块高度 height 以及当前块所处于的 epoch，每个公钥在一个出块周期只能提交一次块，如果重复提交则以最早收到的块为准，其他块则忽略。如果验签失败，则抛弃该公钥所发出的块。

6. SmartX 实验性功能

说明：实验性功能属于探索性质，暂时还不确定发布时间。

6.1. 用惩罚块增加黑名单机制

增加一种交易类型，叫惩罚交易。当全网共识满足一定的条件，由 PoS 决策节点动态投票发送，经过 N 个块高度后，交易生效。惩罚块交易主要用于惩罚恶意攻击，黑客，不诚实节点，惩罚内容包括限制 N 个时间内不允许交易，扣除或者取消手续费等。

6.2. SmartX 应用商店

除了 EVM 虚拟机和智能合约之外，SmartX 将引入一个 RPC 脚本语言，用于丰富 SmartX

的开发生态。在此基础上引入一个类似 Apple AppStore 一样的官方 SmartX 脚本语言程序集散的平台 SmartX store。

6.3. 动态 PoW 哈希算法

使用 Lua/JS 等脚本语言来实现工作量证明的哈希函数，由 PoS 决策节点发起一笔更新算法交易，把算法脚本内容放到交易块中，过程跟动态抵押加入主节点类似。经过一定周期后，新算法生效。

6.4. 加密消息分发

SmartX 的 P2P 网络可以通过节点之间互相协商通密钥，让数据可以在被加密后传输。此外可以通过支持去中心化的存储系统 swarm 和 shh 实现去加密且去中心化的社交应用，而 SmartX 的高性能则为社交应用的体验提供了最可靠的网络保障。

7. 项目路线图

2019/07 - 完成 SmartX 基础框架和基本交易
2019/09 - 完善 SmartX 之选举人投票 PoW 混合共识机制
2020/02 - 上线 SmartX 智图测试网

8. 团队介绍

Frank Su
前腾讯微信支付高级工程师，财付通、微信红包、理财通的早期支付系统核心开发，申请过微信支付两项专利，重构过 XDAG 代码，前以太零 DAG 区块链项目首席工程师。

Bill Ng
前平安保险高级工程师，前 XDAG 钱包开发者，熟悉 Monero 代码，并且对匿名通信有深入的研究。

Li Zhao（技术顾问）
前腾讯微信支付高级工程师，财付通/腾讯移动支付早期核心开发，12 年支付平台经验，精通 BlockDAG 算法，重构过 XDAG 代码。

Jessy
美国斯坦福大学市场营销硕士、美国旧金山艺术大学网页和新媒体设计硕士，拥有丰富的视觉和 UI 设计和产品营销能力。

JunQ
资深运营经理，曾就职于腾讯/乐逗游戏，拥有丰富的团队运营管理经验。

9. 代币分配

- 总量 100 亿
- 43% 社区空投
- 30% 节点挖矿
- 17% 项目开发组（锁定）
- 5% 储备金（锁定）
- 5% 早期生态以及法务（锁定）

10. 参考资料

[1] LWMA Difficulty Algorithm, https://github.com/zawy12/difficulty-algorithms/issues/3.

[2] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters, Twisted Edwards Curves (2008), https://eprint.iacr.org/2008/013.pdf.

[3] Ethash • ethereum/wiki Wiki - GitHub, https://github.com/ethereum/wiki/wiki/Ethash

[4] Ethereum. Ethereum. https://github.com/ethereum/wiki/wiki/White-Paper

[5] IOTA. IOTA. http://iotatoken.com/IOTA_Whitepaper.pdf

[6] Byteball. Byteball. https://obyte.org/Byteball.pdf

[7] Cardano.Ouroboros A Provably Secure Proof-of-Stake Blockchain Protocol. https://iohk.io/research/papers/#ouroboros-a-provably-secure-proof-of-stake-blockchain-protocol

[8] Cardano.Ouroboros Praos – An adaptively-secure, semi-synchronous proof-of-stake protocol. https://iohk.io/research/papers/#ouroboros-praos-an-adaptively-secure-semi-synchronous-proof-of-stake-protocol

[9] J. Chen and S. Micali. Algorand. Technical report, 2017.URL http://arxiv.org/abs/1607.01341

[10] Peercointalk. Peercoin invalid checkpoint.https://www.peercointalk.org/t/invalidcheckpoint/3691, 2015

[11] D. Dolev. The Byzantine Generals Strike Again. J. Algorithms, 3, (1982), pp. 14-30.

[12] D. Dolev and H.R. Strong. Authenticated algorithms for Byzantine agreement. SIAM Journal on Computing 12 (4), 656-666.

[13] P. Feldman and S. Micali. An Optimal Probabilistic Algorithm for Synchronous Byzantine Agreement. (Preliminary version in STOC 88.) SIAM J. on Computing, 1997

[14] Philipp Winter, Roya Ensafi, Karsten Loesing, and Nick Feamster, Identifying and characterizing Sybils in the Tor network (February 25, 2016), https://arxiv.org/abs/1602.07787.

[15] The Sybil Attack.JR Douceur，https://www.freehaven.net/anonbib/cache/sybil.pdf

[16] Go Ethereum - Postal Services over Swarm. https://github.com/ethersphere/go-ethereum/blob/ddfc0a2a02ce574f4c252068ce81f0f5ada1c1ff/swarm/pss/README.md