SDAG设计

修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 修改人 | 修改日期 | 修改内容 |
| V1.0 | 李球 | 2018-09-04 | V0.01版本 |
| V1.0 | 李球 | 2018-09-09 | 添加主链、SDAG架构图 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. 概述

TOS(ThingsOpreatingSystem)是基于区块链技术的全球性智能物联网络开放协议项目，是一种用于物联网IoT行业，基于SDAG的去中心化分层区块网络技术。TOS要做一个普通交易无费用，高价值交易有费用的分层区块网络操作系统。该技术把全网区块形成一个分层的DAG区块网集合，且自动分发海量交易数据到各分层区块网络，大大减少了全网区块数据冗余，同时将网络系统中的数据根据价值标签做分类存储，可以更好地挖掘高价值信息，及未来人工智能结合。这将是区块链技术的下一步延续发展，对区块链具有历史性的创新意义。

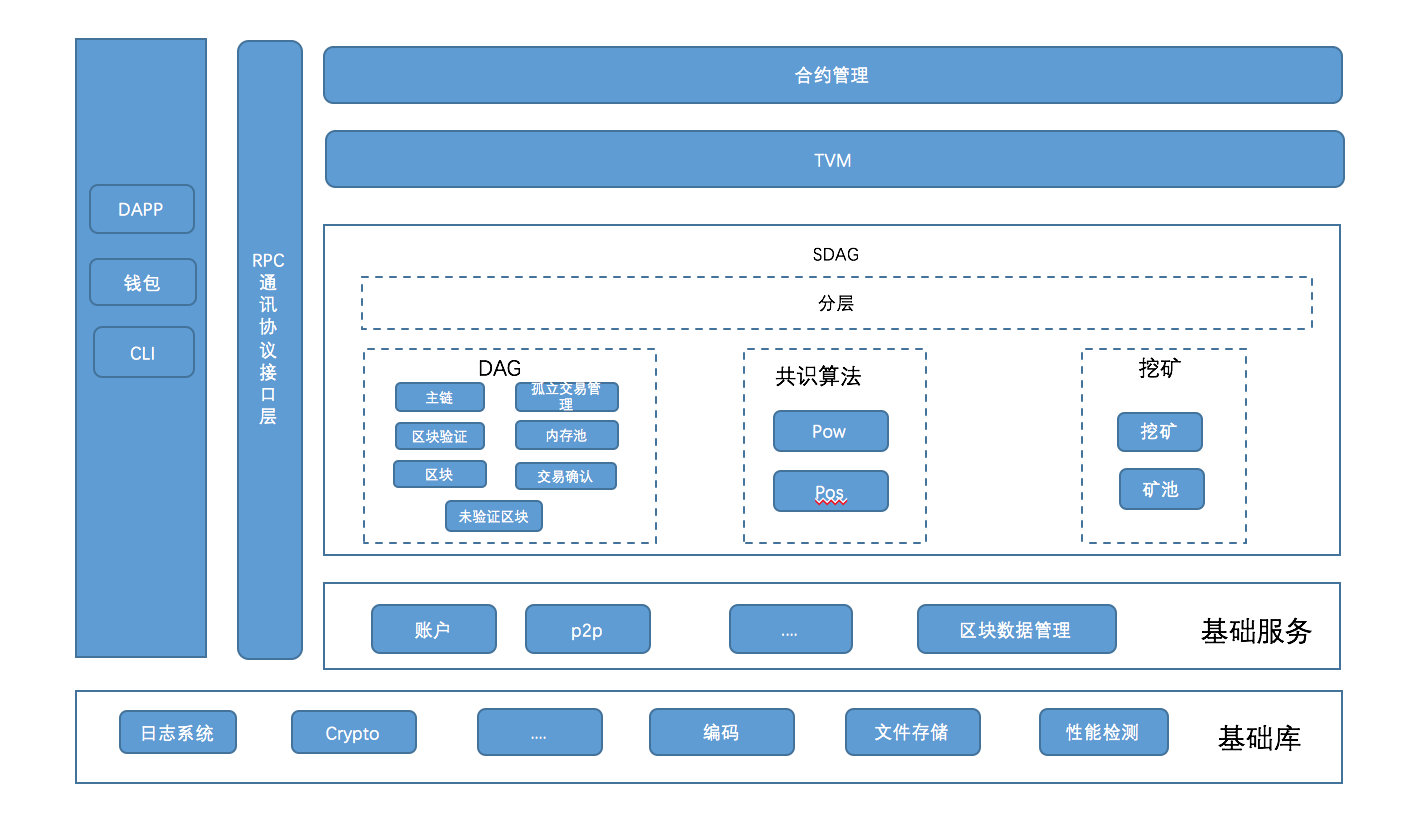
二．目的

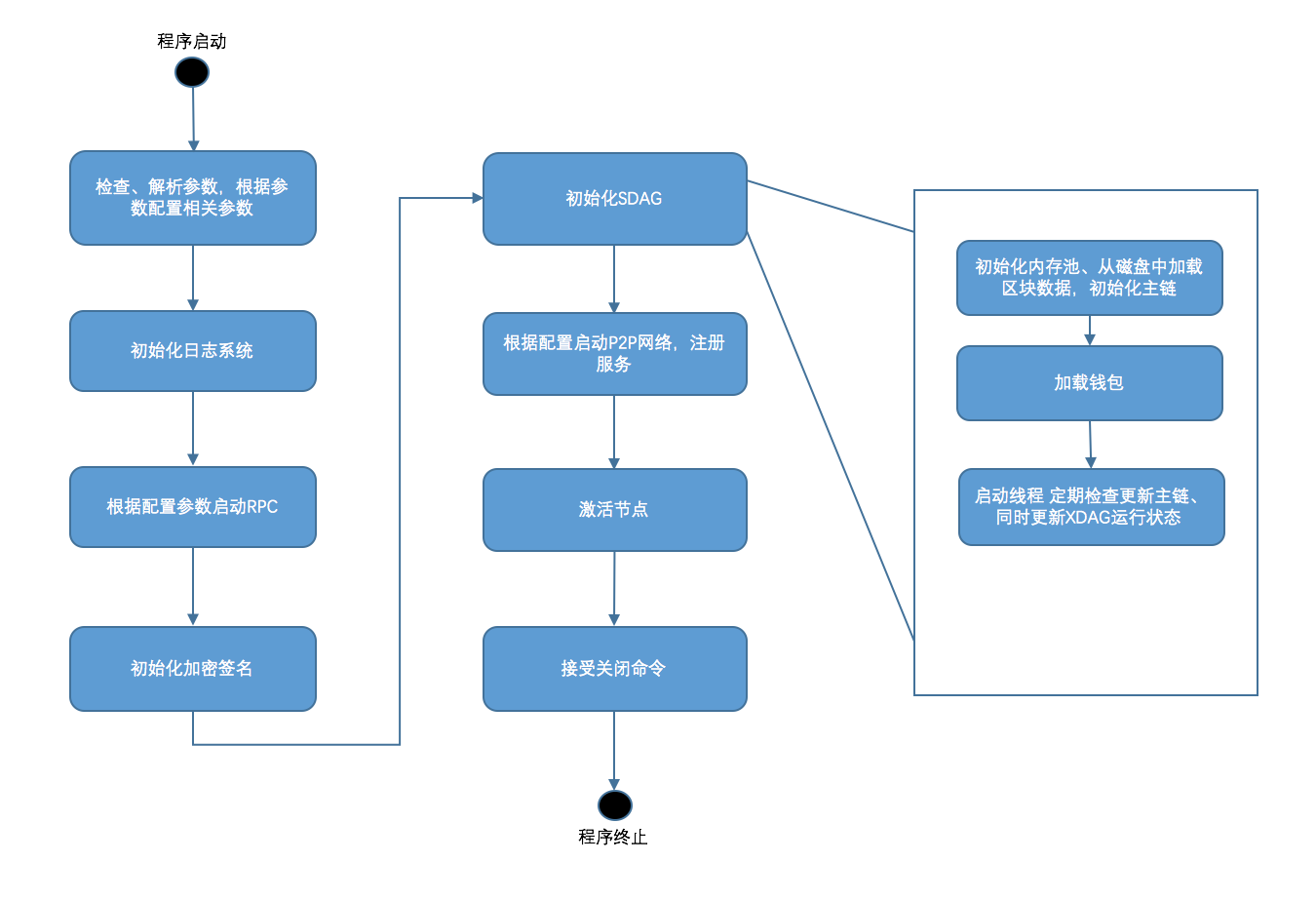
该文档是TOS公有链的概要设计文档，用于指导TOS公链的开发。

三．总体设计

下图为TOS公链的整体架构图，共分为六部分。分别是基础库、基础业务、SDAG、智能合约、RPC接口层、外部应用程序，后面将分章节详细介绍。

3.1 架构图



3.2 启动流程

整个启动过程是解析参数。初始化SDAG，然后创建和启动节点。 再把服务注入到节点中。 所有跟SDAG相关的功能都以服务的形式实现的。

四．基础模块

该模块主要是为上层所有业务提供一些标准的基础方法，分析工具，与业务无关，包含日志系统、密码学相关的工具库、数据编解码、文件存储、性能检测工具、数据类型转换等等。

五．基础服务

该模块是与TOS公链业务相关的基础模块，为SDAG提供一些服务，包括点对点通讯、账户等等

5.1 P2P

p2p通讯网络采用Kademlia算法模型. 采用第三方成熟库.

1. 是否支持黑名单
2. 分叉

5.2 账户

一个钱包地址作为一个独立的交易账户，钱包地址的生成算法见6.1“钱包地址”，需要离线生成。

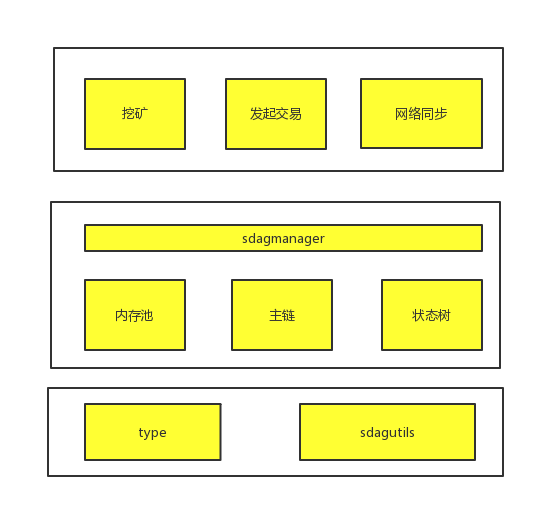
六．SDAG

SDAG大致分为三部分，与区块相关服务、SDAG网、SDAG基础库。

与区块相关服务：如发起交易、挖矿、网络同步，主要是一些区块来源。

SDAG网：是由区块链接关系形成的一张序列的网状结构，所有新建和收到的区块都要添加到SDAG网中，然后达成共识。主链、内存池、状态树、激励都是SDAG网中的组件。

SDAG基础库：SDAG基础数据结构和一些公有的方法.



6.1 名词解释

钱包地址 - 由随机生产256位私钥，再采用secp256k1曲线算法将私钥映射成公钥，然后通过公钥做Keccak-256算法，取最后的160bit。

金额 - TOS的数量，以为tls为最小单位，类似比特币的Santoshi.

金额总量 金额总量上限是10亿TOS。

tls - 一个TOS包括1018个tls. 即 1TOS = 1018tls

区块 - Dagger的基本结构, 结构定义在第二部分介绍.

链 - 一个块的序列，其中每个块被前一个块引用.

Dagger元年 - Unix格式：0x5A500000 ，创世区块上的时间为准.

区块Hash - 对RLP编码的区块数据采用sha3哈希算法, hash(block) = Keccak-256(blockRLP).

区块难度 - (2128-1)/(hash\_little / 2160), hash\_little长度为256 bits(32 bytes).交易区块和无交易区块的hash\_little不相同，交易区块的是区块全部数据RLP编码再Keccak-256，无交易区块是除签名之外的数据RLP编码再Keccak-256。

时间片 - 每216毫秒一个时间片，大约是64秒. 当前区块所在时间片位置 period = block.time >> 16

最大链接数 区块允许的最大的链接区块数。目前为4

链难度 一条链上每个时间片内选取一个区块难度最大的区块，将这些区块的区块难度累加的值。

区块累积难度 从一区块到创始区块的链条数为N，从中找出链难度最大的一条，其链难度就是区块的累积难度。

主链 – 链难度最大的链称为主链.

主块 - 主链上单个时间片内难度最高的区块.

链接 - 如果区块A链接区块组中含有区块B的hash，则区块A链接B，请参考第二部分“区块“ 链接区块组

引用 - 如果从区块A到区块B是一段连续的区块组成，则称区块B被区块A引用

i-引用 - 如果区块A的链接区块组中第i个区块链接到区块C，并且区块C引用区块B，则称区块B是区块A的i-引用.

最短链 - 一个从区块A到区块B的链，每个区块都是i-引用B。i越小，链越短.

区块时间 - 区块时间记录在区块头信息里，从1970年1月1号开始，单位是毫秒.

交易 - 交易即区块.

块最大大小 区块的最大大小

gas 交易费的最小单位，SDAG会按照执行交易的计算量大小收取不同的数量的gas，具体规则待定

gasPrice 单位Gas消耗所折抵的TOS多少

gasLimit 该交易执行过程中所允许消耗资源的总上限，通过这个值，我们可以防止某个交易执行中出现恶意占用资源的问题

6.2 区块

区块有两种，一种是含交易的区块（交易指tos转账或智能合约），另一种是无含交易区块，是用于挖矿。两种区块的难度算法不一致，要特别注意，下面将分别介绍：

6.2.1 交易区块

交易区块是由区块头、交易输出、链接区块组、TVM字节码，VRS签名组成。

1 header 区块头. 结构如下: （是否可以采用RLP）

- 32-bit type 区块种类，这里是1

- 64-bit 区块的时间 单位:毫秒

- 128-bit gasprice. 单位tls

- 64-bit gaslimit. 单位gas

2 链接区块组，以数组的形式存放链接区块信息。存储的区块结构如下。256 \* 4 bit

- 256-bit 区块hash

3 账户交易序号

- 64-bit nonce序号 （账户交易序号， 先预留）

4 交易输出. 结构如下: 384bit

- 256-bit address

- 128-bit 小端字节序, 区块被应用后的金额. 单位tls

5 TVM字节码，是创建智能合约或者执行智能合约方法的TVM字节码

6 R、S、V签名，是将区块中1~4的数据RLP化后，采用ECDSA算法进行签名,将签名的65位长度的签名，分别写入到R、S、V三个字段，R是1 - 32位，S是33-64位，V是最后一位。

区块难度：见6.1 区块难度

6.2.2 无交易区块，即挖矿区块

1 header 区块头. 结构如下: （是否可以采用RLP）

- 32-bit type 区块种类，这里是2

- 64-bit 区块的时间 单位:毫秒

- 128-bit gasPrice. 单位tls

- 64-bit gaslimit. 单位gas

2 链接区块组，以数组的形式存放链接区块信息。存储的区块结构如下。256 \* 4 bit

- 256-bit 区块hash

1. 生成区块的钱包地址

-160-bit 生成区块的钱包地址

1. 随机数

-64-bit 随机数

5 R、S、V签名，是将区块中1~4的数据RLP化后，采用ECDSA算法进行签名,将签名的65位长度的签名，分别写入到R、S、V三个字段，R是1 - 32位，S是33-64位，V是最后一位。

区块难度：见6.1 区块难度

6.3 区块约束

区块需要符合下列要求：

1. 区块是 RLP 格式数据，没有多余的后缀字节;
2. 区块的产生时间不小于Dagger元年；
3. 区块的所有输出金额加上费用之和必须小于TOS总金额;
4. 如果是挖矿区块，需要验证签名是否与区块中所存的钱包地址一致。
5. 挖矿奖励属于区块的签名者。
6. 区块链接的区块是合法的区块;
7. 区块的时间要大于它所链接的区块的时间;
8. gaslimit不小于实际交易过程中用的gas;

区块的来源目前有3种：发起交易、挖矿、网络接受，下面将一一介绍

6.4 发起交易

SDAG区块只能存储一笔交易，意味着当用户想发起一笔交易时就需要创建一个新的区块，大致的流程如下：

1. 创建区块。
   1. 用户将收款方的地址、转出金额、gasPrice和gasLimit信息写入区块中，
   2. 取SDAG中未验证的区块添加到未验证的区块中
   3. 再用自己的私钥对填入的信息进行签名。
2. 将创建好的区块添加到SDAG网中。
3. 添加完成之后发送给其他节点。

示例：地址A转给B 1个TOS, gasPrice设置的120000tls，gasLimit设置40000gas，现在SDAG中有C，D两个未验证的区块，C，D的gas花费是10000gas， 20000gas. 我们需要创建一个区块，

其RLP区块结构如下：

[

[ //header

1, //version

1，//type

1536290816000, //time

2, //nonce

120000, //gasPrice 单位是tls

40000, //gasLimit

],

[

[B\_Address, 1000000000000000000] // B\_Address是B的钱包地址， 1000000000000000000是金额，单位是tls

]，

[ //links 链接区块组

C\_hash , // 链接到C，C\_hash为区块C的hash

D\_hash // 链接到D，D\_hash为区块D的hash

]，

0, //没有智能合约 payload为0

[vsr] //vrs签名

]

6.5 挖矿

SDAG挖矿过程与发起交易非常类似，挖矿也是创建一个新的区块，但区块中与交易相关的数据为空，交易输出。矿工通过改变区块中nonce来生成区块的hash，更可能计算出难度大的区块。流程如下：

1. 创建区块。
   1. 用户header、生成区块的钱包地址写入区块中, header中的time设置为发送时间，即时间片切换的临界时间点 (timestamp >> 16) << 16 | 0xffff
   2. 取SDAG中主链上末端区块hash写入到links中，另外在SDAG中取最大链接数 – 1 个未验证区块hash写入links
   3. 循环累加NONCE，并计算区块的难度，在循环计算期间，每隔1秒检测一下主链是否更新，如果更新，在links中用新主链的末端区块hash替换老的。 后期优化，需考虑新旧主链难度差、区块难度计算时间、及新区块难度值，才决定是否更新link。
   4. 当到达发送时间之后，取难度最大的nonce写入区块
   5. 再用自己的私钥对填入的信息进行签名。
2. 将创建好的区块添加到本地SDAG网中。
3. 添加完成之后发送给其他节点。
4. 进入下一个时间片

区块示例：

[

[ //header

2, //type

1536290816000, //time

120000, //gasPrice 单位是tls

40000, //gasLimit

],

[ //links 链接区块组

C\_hash , // 链接到C，C\_hash为区块C的hash

D\_hash // 链接到D，D\_hash为区块D的hash

]，

sender\_address, //矿工钱包地址

9384793423, //随机数

[vsr] //vrs签名

]

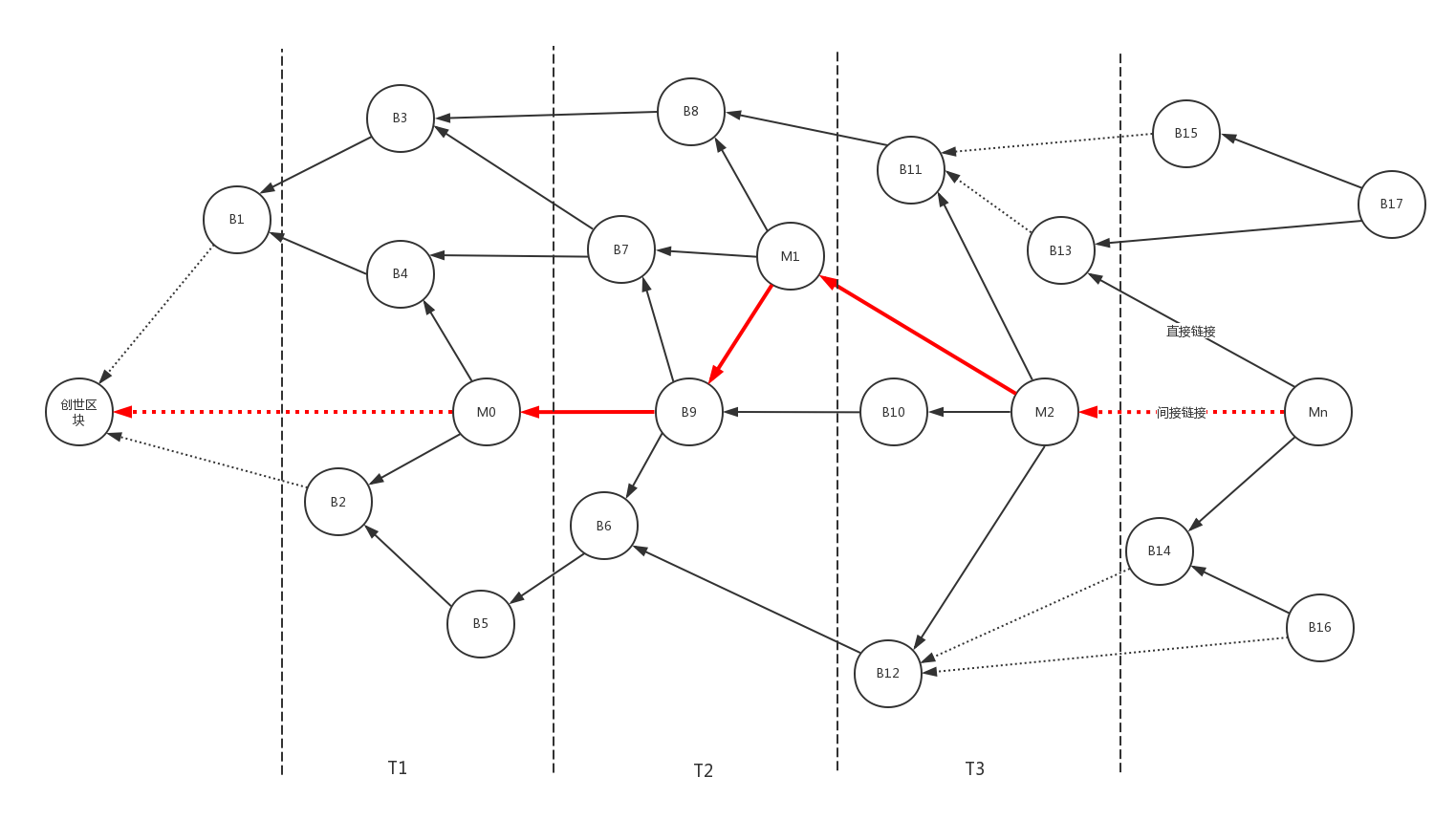
6.6 接收区块

当从网络中收到区块时，我们分为以下几步进行处理

第一步，需要对区块进行验证，因为SDAG比较特殊，我们无法确定数据是否双花，所以只要符合章节6.3所描述的要求，我们都认为是合法区块。

第二步，将区块添加到本地SDAG网中。

6.7 SDAG结构图

­­

SDAG网是区块通过其内部的链接关系所形成的一张网状的拓扑结构图。如上图所示. 是SDAG结构图，由圆圈、带箭头的虚实线、竖直的虚线组合而成。圆圈表示的是区块，区块有三类，创世区块，主块，普通区块。创世区块是全网第一创建的区块，整个SDAG的起点；主块，如图中M0~Mn，一般由矿工创建，也可能是用于做交易的区块，是主链上单位时间片内最难的区块，每个时间片内只有一个主块；普通区块，图中B1-B17，除主块之外的其他区块。带箭头的虚线表示区块之间的间接链接关系，带箭头的实线表示区块之间的直接链接关系。

图中由红色箭头线、创世区块、M0、B9、M1、M2，再到Mn形成一条链，我们称之为主链，该链是整个SDAG网络中累积难度最高的链路，是全网节点通过工作量共同竞争形成的，类似于比特币，主链末的主块不确认，很容易被其他难度更高区块替换，如上图中Mn，如果其所处的时间片内有难度更大的新块B18加入且链到主链上，其累积的难度超过Mn，B18就变成主块，Mn则成普通区块；不过随着时间推移，如果Mn有越来越多的主块链接，这条主链难度越来越大，全网大部分节点也都会基于该条链进行挖矿以期获得奖励，其难度就会更高，如果想改变Mn的主块地位，攻击者就必须分叉出一条累积难度更高的链条，但其算力需达到全网51%，才能有追赶上的可能性，并需要花费大量的时间，其成本巨大，远不如基于现有的主链挖矿收益高，几乎没人会这么干，所以此时的Mn可被认为是稳定，足够安全。

如图，其中M0、M1、M2被后面M3-Mn直接或间接验证，n足够大，认为M0~M2是安全的，它们链接的区块被确认，如图B1~B12被确认。而Mn后面没有主块链接，不够安全，是临时主块，虽然其链接的区块被它验证，但未被确认，如B13~B15。而B16~B17没被任何区块链接，是未被验证的区块。另外假设B3与B4交易冲突，出现双花，根据主块链接的顺序判断B3优先于B4，则B4就是一笔确认但交易冲突的交易，B3是确认且交易合规的区块。

当一个区块同时被多个主块链接，区块总是被靠前的主块确认，如图上B4，同时被主块M0，M1链接确认，由于M0在前，则B4是被M0确认。

6.8 添加区块到SDAG网

不管发起交易、挖矿产生的区块，还是从网络中接受的区块，都是需要添加到本地SDAG网中的，

当一个区块添加SDAG中，流程如下：

1. 需要检查该区块是否已经添加，如添加，略过。
2. 检查links区块是否存在本地，如果找不到，当做孤立区块处理。
3. 检查links区块的时间是否小于该区块的时间，如果不小于，认为是区块不合法，后期考虑将发送该数据的节点加入到黑名单。
4. 将区块存储到内存池中。
5. 将区块添加到主链逻辑，查看主链是否更新。
6. 将新加入的区块添加到未验证区块表中，同时将该区块验证的区块从未验证区块表中移除。

如章节6.7所述，区块添加到SDAG之后，随着时间推移，区块状态会发生变化，可分为6种状态：无状态，被验证，被确认，被应用、临时主块、主块。

|  |  |  |
| --- | --- | --- |
| 状态类型 | 描述 | 值（数据库记录） |
| 无状态 | 区块刚加入SDAG网，无区块链接它 | 1 << 0 |
| 被验证 | 区块已其他区块链接 | 1 << 1 |
| 被确认 | 区块被认为安全的主块直接或间接链接到 | 1 << 2 |
| 被应用 | 区块被确认且其中的交易没有冲突 | 1 << 3 |
| 临时主块 | 主链上且在该时间片内难度最高的区块，未经过32个时间片确认 ，不够安全 | 1 << 4 |
| 主块 | 主链上且在该时间片内难度最高的区块，经过32个时间片确认，认为足够安全 | 1 << 5 |

6.9主链

6.9.1 主链更新

当SDAG网收到一个新的合格区块时，如果难度足够高，以该区块到创世区块能找出一条链路，比目前主链的难度高，就需要更新主链。所有每收到新区块，我们都需要检查。 另外极限情况下，可能会产生分叉，需要把旧主块确定的状态回退。并重新进行区块确认。

分叉处理时需要注意，单位时间片内主链上除了主块，还可能有其他区块，这点需要特别注意。

1. 计算新区块的累积难度，并保存在levelDB中，存储的key: hash\_diff
2. 如果新区块的累积难度比目前主链难度高，将新区块至创世区块形成的链难度最大的链更新为主链，否则退出
3. 更新主链后，检查是否出现分叉，判断标准，该主链上上一个时间片的主块是否已经标记临时主块。如果不是，即出现分叉。
4. 发现分叉，首先新链上的主块标记为临时主块，同时找到分叉点，将原链上有标记为主块的，需将其确认的区块转态回退，将标记为临时主块的标记去除。

6.9.2 区块确认

如章节6.7所述，在主链上的主块被后续足够多的主块验证，会被认为是足够安全的，其直接和间接引用的区块被确认，所有相关交易被确认。之后计算矿工奖励及其他交易数据，处理双花交易。

主块确认条件：

确认的主块需经过后续32个主块的确认，由于一个时间片只产生一个主块，也就是说主块在产生32个时间片之后被确认，后期需要通过测试，来调整该参数。

1. 在主链上查找最早的临时主块（未确认），区块时间是否过去32个时间片，如果没有，退出，否则继续。
2. 先将该区块的状态更新为主块。
3. 根据区块的链接区块组形成的链接关系进行遍历。以该主块为根节点，主块所确定的区块为节点形成的多叉树，做后序遍历，并依次计算转账和交易费用。
4. 计算主块的矿工收益。

6.10 区块持久化存储

内存池是用于缓存区块对象，并自动将区块数据持久化存储到LevelDB. 是一个二级缓存结构。用于减少高频数据编解码。由Memory+ LevelDB实现。有以下几点要求：

1. 添加到内存池的区块数据，需要自动保存到levelDB
2. 查询数据时，memory中没有，自动查询levelDB相应数据，并缓存起来。
3. 缓存过期处理。采用LRU算法。

LevelDB,需要持久化存储的区块数据，即存到LevelDB的：

* 原始RLP区块数据
* 区块状态
* 区块被谁确认
* 累计难度
* 发送者（待定）。

区块原始数据，单独储存，按时间片分类存储

|  |  |  |
| --- | --- | --- |
| 值 | Key |  |
| 区块原始RLP数据 | Hash |  |
|  |  |  |

区块状态数据库 （与账户状态区别）

|  |  |  |
| --- | --- | --- |
| 值 | Key | 描述 |
| 区块状态 | Hash+“s” | 状态见章节6.8 |
| 区块被谁确认 | Hash+“c” | 区块该区块的主块hash |
| 区块的累积难度 | Hash+“d” | 区块的累积难度 |
| 区块的最大难度链接 | Hash+”m” | 区块链接组的序号，从0开始 |
|  |  |  |

6.11 状态树

用于快速存取账户余额，且在主链出现分叉时，实现数据的快速回滚更新。

6.12经济激励

经济激励是为了激励那些按照SDAG共识运行的节点，是公有链长久运行下去的基石，激励设计的好坏，直接影响SDAG的性能。经济激励从以下几个维度考虑：

1. 鼓励参与者基于最难链生成区块，维持主链的稳定
2. 鼓励参与者尽可能多的链接未验证的节点，保证单位时间链接节点的效率

… 待补充

6.13黑名单

某些节点故意发送垃圾数据，和一些不合法的区块，需要将这些节点加入到黑名单。

6.14 区块同步

七．智能合约

待定：SDAG之上建立的合约系统，

八．RPC接口层

该模块的目的是为外部应用程序提供标准的服务接口，为第三方公司接入提供入口。

九．外部应用程序

外部应用程序，是基于TOS公链节点所提供的服务开发的应用程序，主要分为2类，一类是TOS钱包，为用户提供操作TOS的可视化交互界面，另一类，基于TOS智能合约开发的DAPP。