交易分组并行可行性阐述

本方案阐述一种可以通过一种方式将一个区块中打包的交易，进行分组、分批然后并行执行的方式提高区块链中交易的执行效率以及提升区块链的整体效率和速度。下面是对于本方案的详细阐述和示意图。

本阐述方案的实现基础是基于一个已实现的或可实现的技术前提，即合约注册关系表。合约注册关系表登记记录的是一个智能合约中可能调用到的所有非本合约的其他智能合约。这种调用关系可以是一个也可以是多个，由合约的编写者来完成本操作，因为相较于其他人，合约的编写者更加熟悉最新部署的这一智能合约调用到了哪些其他的智能合约。合约注册关系表中的注册关系形式可以是一对一，也可以一对多，并且合约默认对合约自己依赖。

当节点在接收到一个新的区块或触发打块逻辑从交易池中打包一批交易之后，所需要做的动作是统一的，即将交易进行逐一执行。这是区块链的统一行为，而这也正是限制区块链吞吐量的十分重要的难点之一。为了更好地描述本方案的实现细节，预先阐述一批待执行交易的形式如下：

交易哈希	交易发送者	交易接受者
0x0001	Address 1	Address a
0x0002	Address 2	Address b
0x0003	Address 3	null
0x0004	Address 1	Address c
0x0005	Address 1	Address a
0x0006	Address 2	Address 1
0x0007	Address 3	Address b
0x0008	Address 4	Address 3
0x0009	Address 2	Address A
0x00010	Address 1	Address A
0x00011	Address 3	Address B
0x00012	Address 4	Address C

0x00013	Address 5	Address E
0x00014	Address 6	Address G
0x00015	Address 7	Address F
0x00016	Address 4	Address C
0x00017	Address 5	Address H
0x00018	Address 8	Address H
0x00019	Address 3	Address I
0x00020	Address 9	Address erc20 contract X
0x00021	Address 10	Address erc20 contract Y
0x00022	Address 11	Address erc20 contract Z
0x00023	Address 12	Address J
0x00024	Address 13	Address K
0x00025	Address 14	Address e
0x00026	Address e	Address f

其中交易哈希为伪哈希值仅用于区分交易，交易发送者用Address加数字的方式表示，代表普通账户即from。交易的接受者用Address加小写字母表示其他普通账户（用以区别交易发送者from）；用Address加大写字母表示智能合约地址；用Address加erc20 contract加大写字母表示erc20类及token类非本币的智能合约。null表示部署智能合约的交易。

根据交易的接受者类型会发现交易大致可以分为两类：1.账户与账户交互 2.账户与合约交互。其中第二类又可以分成两类：2.1 不涉及token及本币交易的智能合约 2.2 token及本币类的智能合约。

那么可以定义合约注册关系如下：
1.当智能合约中涉及到token、非本币类型的功能时，该智能合约的注册关系为与token有关。
2.当智能合约中涉及本币类的转账类型的功能时，该智能合约的注册关系为与coin有关。
3.当智能合约中涉及到了其他智能合约的功能或与其他智能合约有交互时，但是合约编写者没有手动注册合约的调用关系，那么该智能合约的注册关系为与sync有关。
4.当智能合约中涉及到了其他智能合约的功能或与其他智能合约有交互时，那么该智

能合约的注册关系为与其使用到或间接使用到的智能合约有关，如Address A,Address B等。

现假设：智能合约A中使用到了智能合约B,C;智能合约B使用到了智能合约C;智能合约E使用到了智能合约F,G;智能合约C使用到了智能合约I。合约注册关系表可表示为如下形式：

智能合约地址	合约注册关系
Address A	Address B,Address C
Address B	Address C
Address E	Address F,Address G
Address C	Address I

智能合约注册关系是正向相关，逆向相关性不成立，即Address A与Address B，Address C有关，但是Address B与Address A有关系性推测不成立，Address C与Address A有关系性推测也不成立。

那么根据合约注册关系表，构建出当前这批交易的全交易画像，全交易画像分为两种，

第一种：对交易发送者构建标记画像，即from标记画像。包含交易发送者地址、交易哈希、数字标记三项。

第二种：对交易的接收者构建标记画像,即to标记画像。包含所有合约的地址，发送者地址，token，sync以及coin。

标记画像构建规则如下：

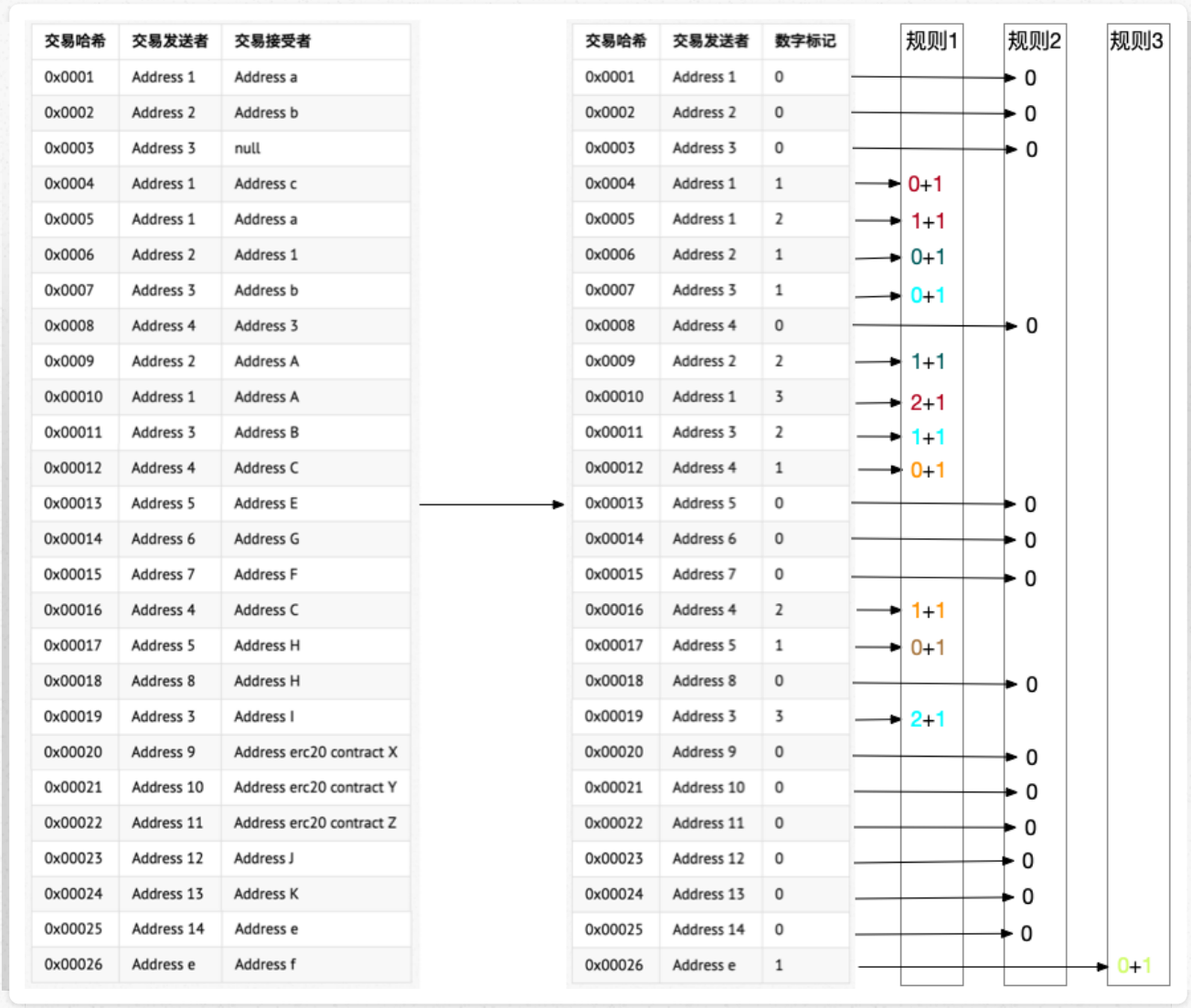
- 对于from标记画像

1.如果是相同的交易发送者不同的交易哈希，那么该条标记画像中的数字标记项按照升序逐笔递增，数字标记从零开始。

2.如果是不同交易发送者不同的交易哈希，那么该条标记画像中的数字标记项统一设置为零。

3.查看交易中的接受者同样为普通账户的交易，如果在from标记画像中的交易发送者地址项找到了，那么就更新这一条标记画像中对应数字标记项数值，进行加一操作。

如下所示：



根据from标记画像的结果，将数字标记为0的交易作为第一批次优先待执行交易。依次根据数字标记的数值作为第几批次优先待执行交易。经过筛选得出如下结果：
第一批次的交易有：
0x0001,0x0002,x0003,0x0008,0x00013,0x00014,0x00015,0x00018,0x00020,0x00021,0x
第二批次的交易有：0x0004,0x0006,0x0007,0x00012,0x00017,0x00026
第三批次的交易有：0x0005,0x0009,0x00011,0x00016
第四批次的交易有：0x00010,0x00019

当初步分出交易批次以后，进行第二个画像的构建。

- 对于to标记画像

  1.首先将所有在本批次交易中调用到的合约地址进行统计，并将所有涉及本币及本币类的合约引申为coin类型；所有erc20token类合约引申为Token类；没有注册合约调用关系但是合约里面调用了其他合约的合约引申为sync类型；将to为空的交易(即部署合约交易)同样引申为coin类型；

  2.将当前批次交易中的当前某笔交易调用到的智能合约中所能涉及到的所有智能合约(无论是直接或者间接的涉及到)标记栏里标记为1，没有涉及到的其他智能合约标

记栏标记0;

如下:

交易发送者	sync	Address A	Address B	Address C	Address E	Address F	Address G	Address H	Address I	Address J	Address K	Coin	Token
Address 1	0	1	1	1	0	0	0	0	1	0	0	1	0
Address 2	0	1	1	1	0	0	0	0	1	0	0	1	0
Address 3	0	0	1	1	0	0	0	0	1	0	0	1	0
Address 4	0	0	0	1	0	0	0	0	1	0	0	1	0
Address 5	0	0	0	0	1	1	1	1	0	0	0	0	0
Address 6	0	0	0	0	0	0	1	0	0	0	0	0	0
Address 7	0	0	0	0	0	1	0	0	0	0	0	0	0
Address 8	0	0	0	0	0	0	0	1	0	0	0	0	0
Address 9	0	0	0	0	0	0	0	0	0	0	0	0	1
Address 10	0	0	0	0	0	0	0	0	0	0	0	0	1
Address 11	0	0	0	0	0	0	0	0	0	0	0	0	1
Address 12	0	0	0	0	0	0	0	0	0	1	0	0	0
Address 13	0	0	0	0	0	0	0	0	0	0	1	0	0
Address 14	0	0	0	0	0	0	0	0	0	0	0	1	0
Address e	0	0	0	0	0	0	0	0	0	0	0	1	0

在完成了对to标记画像的构建以后，将已经拆分完批次的交易按照批次逐个交易与to标记画像中的记录进行检索，然后完成对当前遍历到的交易的批次顺序进行调整，规则如下:

1.如果当前遍历到的交易的接受者出现在了同组交易的发送者位置或者与同组交易的交易接受者有关系的，则说明这两笔交易存在确定的先后顺序无法并行，因此需要将后来这笔与当前遍历到的这笔交易有关系的交易进行降级，即从高优先级分组中向后移动一个分组。

2.如果当前遍历到的交易分组中都是token类型的交易的时候，由于token类的交易合约内部实现较自由，我们无法确定其准确的token流向，因此token类交易都无法并行处理，在同组中token类交易除当前遍历到的交易外，其余的交易依次降低优先级将交易向后移动一个分组。

3.当一笔交易是从高一级别的分组中降低到底一级别的分组中而又与其有直接或间接的关系时，优先进行降级的是原本属于底一级别分组里的交易。

4.如果交易一笔交易并没有注册他的调用关系，但是在实际运行的时候发现了这笔交易除了最外层的接收着外，还调用了其他的智能合约。由于是在运行时发现的调用关系，而且这种关系涉及到的交易充满了不确定性，就讲当前运行到的这笔交易按照交易失败来处理。

因此原批次交易通过变化规则之后结果应为:

第一批:
0x0001,0x0002,0x0003,0x0008,0x00013,0x00018,0x00020,0x00023,0x00024,0x00025

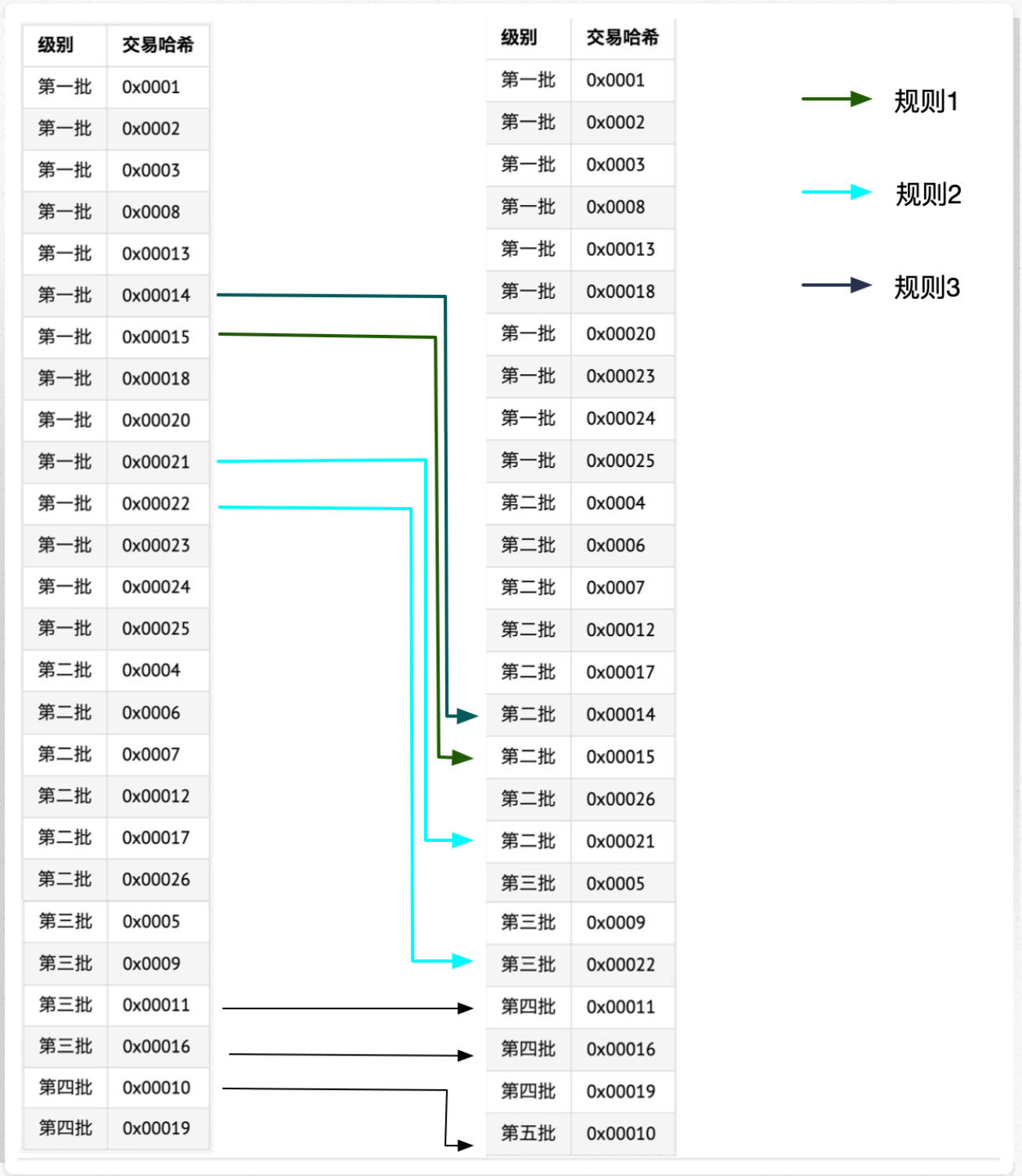
第二批:
0x0004,0x0006,0x0007,0x00012,0x00017,0x00014,0x00015,0x00026,0x00021

第三批：0x0005,0x0009,0x00022

第四批：0x00011,0x00016,0x00019

第五批：0x00010

交易变化情况如图:



同一批次的分组交易可以并行执行，不用批次的交易顺序执行。