



(10) 授权公告号 CN 114041156 B

(45) 授权公告日 2026. 01. 02

(21) 申请号 202080046817.6

(22) 申请日 2020.06.22

(65) 同一申请的已公布的文献号  
申请公布号 CN 114041156 A

(43) 申请公布日 2022.02.11

(30) 优先权数据  
1950780-5 2019.06.25 SE

(85) PCT国际申请进入国家阶段日  
2021.12.24

(86) PCT国际申请的申请数据  
PCT/SE2020/050639 2020.06.22

(87) PCT国际申请的公布数据  
W02020/263162 EN 2020.12.30

(73) 专利权人 数币投资池公司  
地址 瑞典斯德哥尔摩

(72) 发明人 D·杰奎特

(74) 专利代理机构 北京英创嘉友知识产权代理有限公司 11447  
专利代理师 南毅宁

(51) Int.Cl.  
G06Q 20/06 (2006.01)  
G06Q 20/40 (2006.01)  
H04L 9/32 (2006.01)  
G06F 21/60 (2006.01)  
H04L 9/06 (2006.01)

(56) 对比文件  
US 2017366516 A1, 2017.12.21  
US 2019132131 A1, 2019.05.02  
WO 2019032089 A1, 2019.02.14

审查员 罗湘

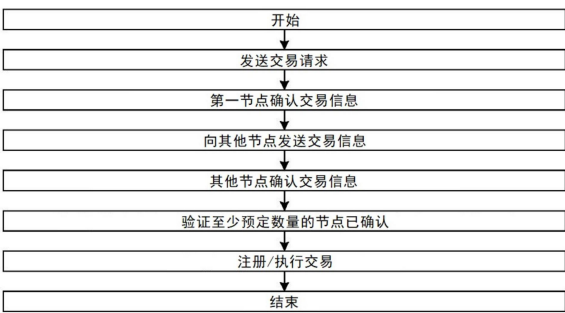
权利要求书2页 说明书24页 附图3页

(54) 发明名称

用于执行电子交易的方法和系统

(57) 摘要

执行电子交易的方法,包括步骤:a) 第一用户客户端(110)以数字方式向第一节点(150)发送交易信息,所述交易信息包括第一用户可预测的交易计数器,所述计数器对于所述第一用户和所述电子交易的组合而言是唯一的,以及用户交易状态摘要,所述用户交易状态摘要是基于相对于所述第一用户注册的先前电子交易计算的单向函数的输出;b) 所述第一节点(150)确认所述交易信息;c) 所述第一节点(150)将所述交易信息以数字方式传送给由此类节点组成的电子交易网络上的其他节点(160、170);d) 所述其他节点(160、170)中的至少一个确认所述交易信息;e) 验证至少预定数量的节点已经确认交易信息;f) 基于所述交易信息注册交易并且以数字方式向所述网络上的所有节点(150、160、170)传播关于至少所述预定数量的节点的肯定确认状态的信息。



1. 用于执行电子交易的方法,包括以下步骤:

a) 第一用户客户端 (110) 以数字方式向第一节点 (150) 发送交易信息,所述交易信息包括第一用户可预测的交易计数器,所述计数器对于所述第一用户和所述电子交易的组合而言是唯一的,以及用户交易状态摘要,所述用户交易状态摘要是基于相对于所述第一用户注册的至少紧接在前的电子交易计算的单向函数的输出;

b) 所述第一节点 (150) 确认所述交易信息;

c) 所述第一节点 (150) 将所述交易信息以数字方式传送给由此类节点组成的电子交易网络上的其他节点 (160、170);

d) 所述其他节点 (160、170) 中的至少一个确认所述交易信息;

e) 所述第一节点 (150) 和所述其他节点 (160、170) 中的一个或多个节点验证至少预定数量的节点已经确认交易信息;

f) 所述第一节点 (150) 和所述其他节点 (160、170) 中的一个或多个节点基于所述交易信息注册交易并且以数字方式向所述网络上的所有节点 (150、160、170) 传播关于至少所述预定数量的节点的肯定确认状态的信息,

其中所述第一节点 (150) 和其他节点 (160、170) 中的每一个包括关于先前第一用户的交易计数器以及关于更新的用户交易状态摘要的信息,并且所述确认所述交易信息包括通过与所述所有节点 (150、160、170) 中的所述关于先前第一用户的交易计数器以及所述关于更新的用户交易状态摘要的信息进行比较,验证所述第一用户的交易计数器是否匹配其预期值,以及验证所述用户交易状态摘要是否匹配所述更新的用户交易状态摘要;以及

其中,所述用户交易状态摘要作为单向函数计算链的最后计算结果来计算,因此,所述所有节点 (150、160、170) 自动为每个用户构建并维护一个由用户交易状态摘要相互连接的独立交易链。

2. 根据权利要求1所述的方法,其中所述第一用户的交易计数器的预期值可以从紧接所述电子交易之前的电子交易中使用所述先前第一用户的交易计数器确定。

3. 根据权利要求1所述的方法,其中所述第一用户的交易计数器在除了所述第一用户之外还存在的多个不同用户之间不是唯一的。

4. 根据权利要求1所述的方法,其中所述用户交易状态摘要被计算为单向函数的输出,所述单向函数是基于相对于所述第一用户注册的最后一次电子交易并且还基于相对于所述第一用户注册的其他先前电子交易计算的。

5. 根据权利要求4所述的方法,其中所述用户交易状态摘要被计算为基于相对于所述第一用户注册的所有先前电子交易而计算的单向函数的输出。

6. 根据权利要求1所述的方法,其中在步骤e中,验证结果是否定的,并且其中所述电子交易因此被拒绝。

7. 根据权利要求1所述的方法,其中响应于步骤f中的传播,所述所有节点 (150、160、170) 注册交易并更新它们各自的关于当前第一用户交易计数器的信息和更新的第一用户交易状态摘要,以在执行与所述第一用户相关的后续电子交易相关的步骤d时使用。

8. 根据权利要求1所述的方法,其中所述方法包括形成经验证的电子交易区块,其中每个区块与特定的唯一区块标识符相关联,并且其中所述交易信息还包括关于与先前形成的所述区块之一相关联的唯一区块标识符的信息。

9. 根据权利要求8所述的方法, 其中至少一个所述区块包括与所述第一用户以及不同的第二用户相关的交易。

10. 根据权利要求8所述的方法, 其中先前形成的所述区块之一不是最后形成的区块。

11. 根据权利要求8所述的方法, 其中所述交易信息还包括关于交易时间戳的信息, 所述交易时间戳由所述第一节点(150)和所述其他节点(160、170)用于确定所述电子交易属于哪个区块。

12. 根据权利要求8所述的方法, 其中所述交易信息还包括区块状态信息。

13. 根据权利要求12所述的方法, 其中所述区块状态信息是基于构成所述区块的一部分的每个经验证的电子交易的相应散列值来计算的。

14. 根据权利要求13所述的方法, 其特征在于, 所述区块状态信息为基于构成所述区块的一部分的每个经验证的电子交易的散列值的异或计算的。

15. 根据权利要求13所述的方法, 其中所述区块状态信息进一步基于先前区块的区块状态信息来计算。

16. 根据权利要求1所述的方法, 其中所述确认所述交易消息进一步包括基于先前存储在所述第一节点(150)和其他节点(160、170)中的第一用户信息, 验证所述第一用户已就所述电子交易获得授权。

17. 根据权利要求1所述的方法, 其中所述交易信息进一步包括第一用户签名, 其中所述确认所述交易消息进一步包括验证所述第一用户签名的真实性。

18. 根据权利要求1所述的方法, 其中所述第一节点(150)执行步骤e中的验证。

19. 根据权利要求18所述的方法, 其中所述所有节点(150、160、170)单独地执行步骤e中的验证。

20. 根据权利要求1所述的方法, 其中所述第一节点(150)和其他节点(160、170)中的每一个都包括一个各自的第一用户操作计数器参考, 用于用户可以针对所述所有节点(150、160、170)发起的一组不同类型的操作, 所述第一用户操作计数器参考中的每一个被配置为参考所述第一用户的交易计数器的特定各自的一个值, 其中步骤f中的所述注册交易包括所述所有节点(150、160、170)根据所述电子交易表示的一个或多个操作更新所述第一用户操作计数器参考中的一个或多个相关参考。

21. 根据权利要求20所述的方法, 其中所述方法包括启动特定节点的步骤, 在所述步骤中, 所述特定节点从与所述特定节点不同的不同节点读取前沿状态, 所述不同节点可能已首先设置为临时只读模式。

## 用于执行电子交易的方法和系统

### 技术领域

[0001] 本发明涉及用于执行电子交易的方法和系统。具体而言,本发明涉及用于执行加密链接的此类交易(例如基于区块链的此类交易)的此类方法和系统。

### 背景技术

[0002] 如今,有许多已知的方式执行加密链接的电子交易。在这里,交易被“加密链接”意味着所讨论的交易是使用加密单向函数密封的,所述加密单向函数是使用先前交易状态作为输入计算的,有效地将连续的交易在一个加密链中相互连接起来,而这个加密链是可以通过数学方式来验证的。这种交易链通常被认为是安全的,并且可以以分布式方式实现,对相关方来说具有高可用性。

[0003] 如果在集团层面收集并签署了一组单独的交易,则该交易链通常称为区块链。

[0004] 这种链式交易的一个主要问题是交易吞吐量。主要协议,如比特币、以太坊和瑞波币(XRP),通常与最大吞吐量有关,这些吞吐量太低,无法为大量的并发用户服务。例如,一些估计认为,比特币区块链只支持每秒10笔交易的规模。

[0005] 交易吞吐量如此之低的一个原因是,链式交易协议必须提供有效的保护,以防止一种被称为“双重消费”的欺诈行为,即特定价值金额从一个消费者同时支付给两个不同的接收者。为了实现这一点,许多区块链协议使用故意耗时或其他困难的任务(如所谓的“工作证明”)与共识算法相结合,保证整个区块链节点网络对区块链的当前状态确实有效的共识。

[0006] 这种工作证明通常会导致高能量的交易成本。

[0007] 除了防止重复消费和提供高的最大交易吞吐量外,链式交易的系统还应该提供短的交易验证和结算时间。例如,在比特币协议中,传统的做法是等待一定数量的区块,如最多六个区块的“开采”,然后才认为交易被完全验证。这可能需要一个小时的时间,这是交易方在最终验证可用之前等待的相对较长的时间。

[0008] 此外,链式交易系统应该为参与的用户提供充分的透明度,并在需要时提供充分的匿名性。

[0009] 这样的链式交易系统还应该是灵活的、可动态实施的和可扩展的,能够根据用户需求的变化而迅速扩展或收缩。

[0010] 除了这些要求之外,在某些情况下,还应该支持超级用户的功能,这样一个受信任的中央方,如中央银行,可以控制个别交易,如阻止欺诈活动和扣押资金,并禁止某些参与者。

[0011] 此外,在发生欺诈或任何其他不当行为的情况下,最好能够安全地回滚链式交易。

### 发明内容

[0012] 本发明的目的是至少解决上述的一些问题。

[0013] 因此,本发明涉及一种用于执行电子交易的方法,包括以下步骤:a)第一用户客户

端以数字方式向第一节点发送交易信息,所述交易信息包括第一用户可预测的交易计数器,所述计数器对于所述第一用户和所述电子交易的组合而言是唯一的,以及用户交易状态摘要,所述用户交易状态摘要是基于相对于所述第一用户注册的先前电子交易计算的单向函数的输出;b)所述第一节点确认所述交易信息;c)所述第一节点将所述交易信息以数字方式传送给由此类节点组成的电子交易网络上的其他节点;d)所述其他节点中的至少一个确认所述交易信息;e)验证至少有预定数量的节点确认了所述交易信息;f)基于所述交易信息注册交易并且以数字方式向所述网络上的所有节点传播关于至少所述预定数量的节点的肯定确认状态的信息。

## 附图说明

[0014] 下面结合本发明的实施例和附图详细说明本发明,其中:

[0015] 图1是可以实施根据本发明的方法的系统的概览图;

[0016] 图2是说明根据本发明的方法的流程图;

[0017] 图3是说明同样根据本发明的方法的流程图;

[0018] 图4说明了连续区块之间的关系;和

[0019] 图5是说明同样根据本发明的方法的流程图。

## 具体实施方式

[0020] 图1示出了其中可以实现根据本发明的方法的系统100。系统100的各种组件经由因特网101或经由任何其他合适的数字广域网互连。

[0021] 系统100包括多个计算机客户端110、120、130,每个客户端是物理通用可编程计算实体,例如个人计算机、平板计算机或智能手机;或虚拟通用可编程计算实体,例如分布式或虚拟化虚拟计算资源。每个计算机客户端110、120、130被配置为例如在所述客户端110、120、130的硬件上或从所述客户端110、120、130的硬件执行客户端软件功能,所述客户端软件功能又被配置为根据本方法执行为计算机客户端110、120、130规定的步骤。例如,此类功能包括与其他实体的通信、数据处理/计算和存储。

[0022] 每个计算机客户端110、120、130通常被配置为允许一个特定用户访问系统100,并且特别是请求/执行将在下文中描述的数字交易。每个计算机客户端110、120、130还可以被配置为允许用户访问和编辑来自下述账户服务器140的用户账户信息。就此而言,每个计算机客户端110、120、130可以被配置为向所讨论的相应用户提供交互式用户界面。例如,这样的用户界面可以是在所讨论的计算机客户端110、120、130的屏幕显示上呈现的交互式图形用户界面。系统100还可以为自动化用户提供服务,并且在这种和其他情况下,每个客户端110、120、130可以提供诸如API(应用程序接口)之类的数字接口,通过该接口接受命令等等。

[0023] 包含在计算机客户端110、120、130中或计算机客户端110、120、130被配置为在其上运行的计算机硬件通常包括至少一个CPU、至少一个RAM存储器和至少一个计算机总线。

[0024] 系统100还包括计算机账户服务器140,其可以是如关于所述计算机客户端110、120、130所描述的相应类型。因此,账户服务器140可以是物理的或虚拟的,并且被配置为运行特定账户服务器软件功能,进而被配置为执行通过本方法为账户服务器140规定的方法

步骤。

[0025] 一般而言,账户服务器140被配置为保持所述用户中的至少一个的当前资源账户。这样的账户可能是金融账户,持有任何货币的某种类型的价值,包括数字货币,或任何类型的所有权份额。账户服务器140因此可以属于银行。然而,在优选实施例中,账户是系统100内部的,在这个意义上,账户服务器140(包括持有的账户)是系统100的集成部分,不提供第三方可以访问账户信息的任何外部接口。例如,帐户服务器140可以是每个节点150、160、170的集成功能部分,以便在没有节点对如下所述的改变进行投票的情况下不允许账户改变。

[0026] 帐户信息保存在数字数据库141中。

[0027] 系统100还包括与所述计算机客户端110、120、130具有相应构造的多个计算机交易节点150、160、170。因此,节点150、160、170每个都与相应的节点软件功能相关联(以对应于已经描述的关于计算机客户端110、120、130的方式),被配置为相对于所述节点150、160、170执行本方法步骤。特别地,节点150、160、170负责跟踪、验证和注册当前类型的数字交易。计算机客户端110、120、130中的每一个包括各自的硬件和/或软件数据库151、161、171或与之接触,每一个进而持有关于数字分类账的信息,所述交易在数字分类账注册,这将在下文中描述。每个这样的数据库151、161、171还可以包括关于系统100中的注册用户的信息,包括用户标识信息和用户状态信息(见下文)。

[0028] 系统100可包括至少5个、优选至少10个所述类型的互连节点,每个节点参与下文描述的针对单个交易请求的投票过程。在各个节点150、160、170和各个用户之间可能没有一对一的关联。换句话说,所有节点可以服务于系统100的所有用户。然后,对于每个节点150、160、170,可以有多个用户,例如至少10个用户,或者甚至每个节点至少100个或至少1000个用户。

[0029] 每个节点150、160、170可存储至少一个用户、或至少一些用户、甚至所有注册用户的部分或完整的分类账。具体地,每个节点150、160、170可以为作为当前类型交易的一方的每个用户存储部分(“前沿状态”(frontier state),见下文)或完整的分类账。对于为其存储信息的每个用户,每个存储的分类帐可以包括从最近注册的交易开始计数并且在时间上倒退直到某个相应最早存储的交易的完整交易集。下面更详细地讨论部分分类帐的情况。

[0030] 一般而言,每个节点150、160、170将数字分类账和关于向系统100注册的用户的附加信息保存在各自的数字副本中,并且这些单独的副本作为下面描述的交易处理过程的自动结果而保持同步。一些节点可能需要持有整个账本(整个交易历史),而其他节点可能只需要持有更近期的交易集(例如账本的“前沿状态”,见下文)。

[0031] 节点150、160、170一起形成“节点网络”,这些节点可以在公共互联网上互连并且可以被配置为使用明确定义的接口和使用加密信道彼此通信以避免窃听。

[0032] 一般而言,这里所说的关于计算机客户端110、120、130的所有内容同样适用于所述客户端软件功能。账户服务器/账户软件功能,以及交易节点/节点软件功能也是如此。所述实体110、120、130、140、150、160、170中的每一个都配备有合适的数字通信接口,用于与其他系统100内部实体以及视情况与外部实体进行通信。这里描述的所有通信都可以被加密,例如基于PKI(公共密钥基础设施)加密。此外,所有通信可以是所涉及的各个实体之间异步的、基于消息的通信。因此,每个实体可以例如实现消息队列,按顺序处理每个传入消

息,导致发送某些动作和/或进一步的消息(或响应消息)。

[0033] 可能有几个不同的账户服务器140,每个都为多个用户提供服务。如上所述,每个节点150、160、170可以包括账户服务器140功能。此外,可以有任意数量的计算机客户端,例如至少10个计算机客户端、至少100个计算机客户端或甚至至少1000个计算机客户端。

[0034] 每个注册用户可以使用所述客户端110、120、130中的任何一个,并且可以通过本身的常规登录或类似方式向系统100(例如向其发送交易信息的节点150、160、170)标识。

[0035] 图2图示了根据本发明的用于执行电子、数字交易的方法。

[0036] 该方法开始于第一步骤。

[0037] 在随后的步骤中,第一计算机客户端110以数字方式向第一交易节点150发送包括交易信息的交易请求。该交易信息可以是第一客户端110发送到第一节点150的消息的形式,该消息包含一组定义交易的信息,第一计算机客户端110的用户(“第一”用户)希望注册和执行该交易。

[0038] 交易可以是金融交易,例如定义要转入或转出账户服务器140持有的第一用户账户的特定货币价值。然而,该交易还可以定义与第一用户和/或第一用户操作的第一计算机客户端110相关的任何其他状态变化,例如第一用户个人信息更新交易;第一用户密钥更改或更新交易;或第一用户添加或用户删除交易(例如在超级用户的倡议下,将特定用户作为活动用户从系统中删除)。有关示例,请参见下文。

[0039] 因此,如在此使用的,术语“交易”将被广义地解释为数字定义的、电子通信的系统100相对于至少一个识别用户的状态改变。这种交易通常由系统100(特别是由节点150、160、170)验证,在所述分类账中注册并执行。执行可以完全以系统100的内部方式发生,例如通过更新存储在数据库151、161和/或171中的识别用户的信息或状态信息;或涉及外部各方,例如涉及外部资金账户的资金转移指令。

[0040] 在优选实施例中,账户服务器140是系统100的一部分,并且所有涉及价值转移的交易都与账户服务器140持有和定义的账户相关地执行,由此这里描述的交易执行不涉及任何外部价值账户持有实体的激活。换句话说,由于当前类型的交易而转移的价值在这种情况下仅在系统100的注册用户之间通过客户端110、120、130提供的接口执行。此外,在这种情况下,可能会定义其他类型的交易,涉及外部实体,例如外部价值存款和取款交易。

[0041] 从第一客户端110发送到接收节点150的交易信息优选地向系统100完整地定义了期望的交易。

[0042] 具体而言,所述交易信息包括第一用户可预测的交易计数器,该计数器对于所述第一用户和所述交易的组合是唯一的。计数器“唯一”意味着由特定用户发起的两个交易不相同。然而,在一些实施例中,用户交易计数器在系统100的几个不同用户之间不是唯一的,除了请求由所讨论的交易信息定义的交易的第一用户之外,还存在不同的用户。

[0043] 计数器是“可预测的”意味着接收节点150可以明确地预测所讨论的交易计数器的预期值,仅使用由同一用户发起的至少一个已知的先前交易信息和/或注册交易(例如交易信息和/或前一交易的注册交易),特别是使用此类先前交易信息的相应交易计数器值。例如,对于每个单独的用户,交易计数器可以是各自的简单单调递增的数字计数器(1、2、3……)。

[0044] 交易信息还包括用户交易状态摘要。用户交易状态摘要可以是基于相对于同一第一用户注册的至少一个先前电子交易计算的单向函数的输出,优选地基于多个或甚至所有此类先前电子交易,在许多情况下,至少基于相对于同一第一用户注册的紧接在前的交易。此外,用户交易状态摘要可以被计算为单向函数的输出,该单向函数是基于相对于请求的第一用户注册的最后一次交易并且还基于相对于相同的第一用户注册的其他先前注册的交易计算的。例如,每个用户状态可以作为单向函数计算链的最后计算结果来计算。在该示例和其他示例中,可以部分地基于先前用户状态摘要的值来计算每个用户状态摘要。

[0045] 单向函数可以是任何合适的单向函数,在计算单向函数的结果时需要相对较少的计算资源,但实际上哪个单向函数的结果并不足以确定所述单向函数的输入。这样的单向函数是众所周知的,在此将不作进一步详细描述。

[0046] 然而,用户交易状态摘要也可以是基于相对于所述用户注册的先前交易的各自散列值计算的摘要,例如所有此类先前注册的交易。然后可以使用预定函数来计算这样的散列。然后,用户交易状态摘要可以是相对于所述用户注册的每个所述先前交易的各自散列的简单异或(XOR),可以使用先前XOR值和新交易的散列相对于每个新交易更新该XOR值。

[0047] 可以理解,该示例中的第一节点150是接收节点。然而,所述节点150、160、170中的任何一个理论上可以由所述用户中的任何一个发起的来自所述计算机客户端110、120、130中的任何一个的交易请求的接收节点。不同的交易请求可以发送到不同的接收节点150、160、170。

[0048] 在随后的步骤中,第一/接收节点150确认所述交易信息。该确认可包括验证所述交易信息中的单个信息片段中的至少一个,优选多个。这些验证可以与第一节点150存储(例如存储在数据库151中)的数据有关。特别地,可以至少部分地相对于存储在所述分类帐中的信息和/或存储在每个节点150、160、170中的用户相关的同步文件进行所述验证(参见下面的示例)。

[0049] 在确认不成功的情况下,例如所述验证中的至少一个结果是否定的,第一节点150将立即拒绝交易请求(“快速拒绝”),将失败消息返回给请求客户端150,而不将交易信息发送给任何其他节点160、170。

[0050] 否则,在随后的步骤中,第一节点150将在所述交易请求中从第一客户端110接收的交易信息以数字方式传送到这些节点的电子交易网络上的其他节点160、170。在优选实施例中,所有节点150、160、170至少以潜在的“赞成”(yes)投票者的身份参与每个交易请求。也就是说,所述其他节点160、170和第一节点150一起构成所述网络上的所有节点150、160、170。

[0051] 在随后的步骤中,其他节点160、170中的至少一些,优选地所有其他节点160、170,每个都对从第一节点150接收的交易信息执行肯定或否定确认,例如执行与第一节点150在前一步骤中执行的相同的验证。

[0052] 对于所述其他节点160、170中的每一个,确认的结果是肯定的(所有验证都是肯定的)或否定的(至少一个验证是否定的)。可以将结果返回到第一节点150以验证来自节点150、160、170的肯定确认的总数。

[0053] 因此,在随后的步骤中,在本示例性实施例中,所述其他节点160、170中的至少一个如上所述确认交易信息。

[0054] 然后,在随后的步骤中,验证至少预定数量的所述第一节点150和其他节点160、170已经确认交易信息。预定数量可以是绝对数量,或者网络中节点150、160、170的特定百分比,例如大于50%的百分比。在一些实施例中,预定数量是节点总数的至少60%,或者甚至是必须报告交易信息的肯定确认的节点150、160、170的至少70%,以便实现至少预定数量的节点已经确认交易信息的肯定验证。在本示例中,后者意味着三个节点150、160、170中的至少两个必须报告肯定的交易信息确认。

[0055] 需要注意的是,一旦接收到至少预定数量的节点已经确认交易信息的信息,该方法就可以进行下一步,而不必等待所有节点150、160、170报告他们的发现。例如,延迟报告可能是由于某些节点停机、通信问题等造成的。

[0056] 在一些实施例中,所有其他节点160、170将所述确认的结果报告给一个特定节点,例如第一(接收)节点150。然后就是这个特定的节点对确认进行验证。这提供了一个快速、灵活的系统100。然而,在许多实施例中,确认的验证发生在几个不同的节点处,例如单独地在所有节点150、160、170处。这样的单独确认然后可以在每个这样的节点150、160、170中独立地发生,并且交易然后也在每个这样的节点中注册。如本文所用,与交易相关的术语“注册”可包括执行交易,例如更新账户服务器140数据,以及更新公共交易分类账,包括区块状态和用户状态(见下文)。

[0057] 如果验证结果是否定的,则交易因此被拒绝,并将拒绝结果反馈给请求客户端110,例如由验证节点150。

[0058] 另一方面,如果验证结果是肯定的,则在随后的步骤中,验证节点150基于所述交易信息来注册交易。这种注册可以包括用数字交易更新账本(存储在数据库151中)的本地副本,然后提交该交易。交易的注册还可以包括交易的执行,例如根据交易信息更新用户信息或基于交易信息结算金融交易。

[0059] 验证节点150以数字方式向所述节点网络上的所有其他节点160、170传播关于与至少所述预定数量的节点的交易信息相关的肯定确认状态的信息。

[0060] 其他节点160、170然后也可以在它们各自本地存储的信息集中注册交易信息,包括它们各自本地存储的分类账副本。这样,分类账将在节点150、160、170之间保持同步。

[0061] 具体地,响应于来自接收节点150的经验证的肯定(高于阈值)投票的所述传播,所有节点150、160、170可以注册交易并更新其关于当前第一用户交易计数器和第一用户状态信息的各自本地信息。然后,该更新信息可在执行与针对同一第一用户请求的后续交易相关的所述确认时使用。

[0062] 关于交易的执行,特别是在信息更新类型交易的情况下,可以针对每个节点本地存储的信息执行(就信息更新而言)。如果执行需要与所述节点网络外部的实体交互,例如涉及账户服务器140的账户的金融交易,则执行仅执行一次。在后一种情况下,验证节点150可以执行交易,或者集中布置的监控服务可以监控分类帐并执行在分类帐上登记的每笔金融交易。

[0063] 在随后的步骤中,该方法结束。

[0064] 这样,实现了用于处理、注册和执行数字交易的非常灵活的方法和系统100。用户可以使用任何节点来请求交易,因此单个接收节点跨节点网络传播接收到的交易信息。然后自动执行上述非常简单的投票算法,如果通过确认进行肯定投票,交易可以立即注册并

执行,延迟时间最短。投票算法还提供了防止双重消费的强大保护,下面提供的更多具体示例将清楚地表明这一点。

[0065] 由于所有参与节点150、160、170可通过定义每个请求的交易的相应交易信息到达,因此每个节点150、160、170都可以始终保持分类账的更新副本,结果在节点网络中可以是相同的,至少除了最近处理的一个或多个交易之外是相同的。在一些实施例中,节点150、160、170可以仅在其“前沿状态”方面不同(参见下文)。因此,全局交易历史的视图始终在整个节点网络中保持明确定义和充分同步(节点至少处于相互不矛盾的状态)。由于本文所述的机制,一个节点注册的交易也将很快被其他节点注册。在某种意义上,分类账的真实状态由节点150、160、170之间的多数票决定。

[0066] 关于同样在节点150、160、170之间保持同步的用户信息文件,相应的情况也是如此。

[0067] 本发明的一个非常重要的方面是,由所述用户交易状态摘要互连的单独交易链由所有节点150、160、170为每个单独用户自动构建和维护,该用户交易状态摘要链独立于涉及多个不同用户的交易的任何区块链结构(见下文)。这种特定于用户的链结构使得能够以非常快速的方式处理每个请求的交易,而不会失去可靠性或安全性。该用户个人交易链本身可以被视为一个轻量级区块链,即使构成这个轻量级、特定于用户的区块链的个人交易也可能构成全球区块链的一部分(见下文)。

[0068] 如上所述,第一用户交易计数器的预期值可以从紧接在当前请求的第一用户相关交易之前的另一个第一用户相关交易中使用的先前第一用户交易计数器中确定性地确定(并且实际上也确定)。在一些实施例中,节点网络上的每个节点150、160、170包括关于所述先前第一用户交易计数器的信息。然后,交易信息的所述确认包括通过与存储的先前第一用户交易计数器进行比较来验证在接收到的交易信息中提供的第一用户交易计数器与第一用户交易计数器的期望值匹配。

[0069] 节点网络中的每个节点150、160、170可以包括关于更新的用户交易状态摘要的信息。然后,在每个单独节点150、160、170中对定义所要求的交易的交易信息进行的本地确认可以包括验证在所讨论的交易信息中提供的用户交易状态摘要与所讨论的确认节点150、160、170中的所述更新的用户交易状态摘要匹配。

[0070] 在非常优选的实施例中,系统100和节点150、160、170被配置为在区块链上注册所述类型的交易。区块链是众所周知的,这里不再详细解释。然而,需要注意的是,区块链是通过在交易区块中收集交易集而形成的,每个区块包含先前区块的单向函数摘要(例如哈希),因此随着时间的推移,形成一个易于验证的交易区块链。

[0071] 即,该方法在这种情况下还可以包括形成这样的经验证和注册的电子交易区块,每个区块与特定的唯一区块标识符相关联。然后,所述交易信息还可以包括关于与先前形成的所述区块之一相关联的唯一区块标识符的信息。区块标识符是“唯一”的,意味着它是系统100全局唯一的,因此没有两个区块具有相同的标识符。例如,每个形成的块可以具有确定性和单调递增的数字系列中的数字作为唯一标识符,例如简单的数字系列1、2、3、4等。

[0072] 优选地,交易信息被要求包括关于区块的唯一区块标识符的信息,该区块与所请求的交易相关地被明确定义,例如基于当前形成的注册交易区块。

[0073] 在一些实施例中,系统100被安排为形成全局区块,在这个意义上,形成的区块通

常包括与不同用户有关的交易,在本例中,与第一用户有关,也与一个或几个不同用户有关,例如至少也与使用第二客户端160请求数字电子交易的第二用户有关。

[0074] 在一些实施例中,先前形成的、其唯一区块标识符包含在交易信息中的所述区块之一不是最后形成的区块。例如,根据交易信息,有可能明确地识别一个区块,在它被注册的情况下,所请求的交易将构成其中的一部分。这种识别可以基于关于目前正在形成的区块的系统全局信息。然后,每个交易信息可能被要求包括一个有效的唯一区块标识符,该标识符标识是在预定数量的区块之前的区块,例如不是最后形成的完整区块,而是在最后形成的完整区块之前形成的区块(见图4)。然后在由所述节点网络中的每个节点150、160、170执行的所述交易信息确认中验证该信息。

[0075] 在一个特定示例中,交易信息还包括关于全局的、优选地在节点150、160、170之间同步的交易时间线的信息,该信息被第一节点和其他节点用来确定电子交易属于哪个区块。例如,全局交易时间线可以是自1970年1月1日以来的毫秒数,由每个节点150、160、170独立使用,结合特定的预定区块形成/关闭时间频率,例如每30秒一次,以确定当前区块,并因此也确定在时间上向后预定数量区块(例如两个)的特定已关闭区块。

[0076] 在一些实施例中,发送客户端110可以使用该全局交易时间线对有关交易进行时间标记,然后处理节点150、160、170可以只验证该时间标记,目的是如果该时间标记与基于有关其他节点的内部时钟的预期时间标记相差超过预定的秒数,则投票“反对”(no)。

[0077] 因此,每个区块可以收集在特定预定时间段内请求的所有交易,作为发送客户端110、120、130提供的每个相应交易上的时间戳。因此,在这种情况下,每个区块不包含预定数量的交易,而是由所述公共时间线定义该区块。时间段可优选为最多1分钟、优选为最多10秒、优选为最多5秒。

[0078] 此外,定义所述交易请求的交易信息可以包括一条区块状态信息或区块状态摘要,例如与也包括在交易信息中的由所述唯一区块标识符标识的相同的先前区块有关。

[0079] 以与上面讨论的用户交易状态摘要相对应的方式,可以将区块状态信息计算为单向函数甚至单向函数链(如上文针对用户交易状态摘要所述)。它也可以是一个XOR类型的摘要,根据每个已验证电子交易(也与其他用户相关)的各自明确定义的哈希值计算,形成所讨论的被引用区块的一部分,例如所述哈希值的XOR。

[0080] 区块状态信息也可以根据区块状态信息和/或构成先前区块(如紧邻的区块)一部分的每笔经核实的电子交易的相应明确的哈希值来计算(如使用单向函数或所述类型的XOR型摘要)。这样,可以构建区块链,其中每个区块的区块状态是基于所有先前区块计算的,特别是基于构成所有先前区块的一部分的所有先前验证的电子交易。

[0081] 除了上述用户交易计数器的验证;用户交易状态摘要;以及块状态摘要,可以包括在由节点网络中的每个节点150、160、170执行的交易信息的确认中的其他验证包括验证请求的第一用户相对于电子交易被授权。这种授权可以基于先前存储在第一节点150和其他节点160、170中的第一用户信息,并且可以,例如,关于第一用户是否被系统100授权执行请求的交易所属的交易类型。这种授权又可以基于由网络中的每个节点150、160、170存储的用户类型信息。

[0082] 此外,这种其他验证还可以包括验证包含在交易信息中的第一用户签名的真实性。即,在向第一节点150发送交易请求之前,第一用户可能需要使用私钥(例如公私PKI密

钥对中的私钥)对交易信息进行签名。然后,由所有节点执行的确认可以包括使用第一用户的对应公共(已知)密钥来验证签名实际上是使用所述私人密钥做出的。

[0083] 在交易信息被传播到其他节点160、170之前,许多这样的检查可能会导致接收节点150已经快速拒绝。

[0084] 同样,如果上述验证中的任何一项失败,则对该节点的确认是否定的。

[0085] 在一些实施例中,所述第一节点150和其他节点160、170(例如所述网络中的所有节点)中的每一个中的都包括一个各自的第一用户操作计数器参考,用于用户可能发起的与所述节点有关的、与执行请求的相同第一用户有关的一组不同各自的不同类型的操作。在该上下文中,每个这样的操作通过请求相应类型的并且如上所述的相应交易来发起。这种操作/交易类型的示例包括“向/从账户X转移资金”、“更新用户公钥”和“更新用户识别信息”。

[0086] 然后,所述第一用户操作计数器参考中的每一个被配置为参考所述第一用户交易计数器中相应的一个的特定相应值。优选地,每个这样的用户操作计数器参考第一用户交易计数器的特定一个存储值,该值标识同一第一用户的特定交易类型的最后注册交易。

[0087] 然后,由所有节点150、160、170执行的交易的所述注册还可以包括所讨论的节点更新所述第一用户操作计数器参考(由所述节点存储)中的一个或多个相关参考,这取决于所述交易所代表的一种或多种操作的类型。因此,在该示例中,用户操作计数器参考集不用于交易确认,而是仅在所讨论的交易的最终验证之后作为每个交易的注册的结果而更新。

[0088] 一旦这样的用户操作计数器引用被存储在节点150、160、170中,它们就可以用于非常快速地恢复故障节点。这种故障节点恢复方法如图3所示。

[0089] 在可以在图2所示的方法结束之前执行的第一步中,该方法开始。

[0090] 在随后的步骤中,例如由所述节点网络中的至少一个节点150、160、170确定特定节点已经发生故障。此类故障可能包括特定节点变得无响应、被检测为故障或被黑客攻击(参见下面的示例)或类似情况。检测可以由任何其他节点进行。

[0091] 在随后的步骤中,特定节点被启动,如新安装和/或重新启动。此时,所讨论的节点可能是先前检测到其故障的恢复节点。然而,对于新启动的节点,可以如下所述执行相同的方法步骤。

[0092] 在随后的步骤中,特定节点读取先前存储且优选地(但不一定)更新的第一用户交易状态摘要,以及先前存储且优选地(但不一定)更新的区块链状态摘要。这种读取可以从任何其他节点执行,也可以从特定节点自身在失败之前所做的先前存储的备份中执行。读取的信息可以包括下面在示例7中描述的一种或几种信息类型(“前沿状态”)。

[0093] 与此相关,特定节点还读取向系统100注册的特定一个、多个或所有用户的更新的第一用户操作计数器参考集。该读取从所述网络中的节点150、160、170中的不同一个执行,该节点首先被设置为临时只读模式。

[0094] 因此,所述特定节点可从可能已首先以临时只读模式设置的不同节点读取以下定义类型的前沿状态。更一般地,所述特定节点可以从可能首先以临时只读模式设置的不同节点读取最小化状态,该最小化状态比整个分类账小得多,但仍然足以继续操作(最小化状态可以是所述前沿状态)。

[0095] 在随后的过程中,特定节点可使用所述第一用户操作计数器参考集进行确认,以

验证所述特定节点存储的第一用户信息是最新的。该验证可以基于每个单独用户操作计数器参考的值来进行,确定用户特定信息在交易历史中的多远之前必须是完整的。如果在此类检查之后有必要,特定节点可以从任何其他节点请求更新的此类第一用户信息。

[0096] 因此,在所述节点网络的每一个节点150、160、170的各自相同副本中持有、维护和更新的分类账是由不同的、受信任的、无层次结构的节点150、160、170运行的分类账。如上所述,这些节点150、160、170通过投票批准或拒绝操作。如果特定数量或百分比的参与节点150、160、170(例如合格的大多数节点)批准操作,则所有节点将提交该操作。否则就不行。与许多传统的区块链加密货币不同,每个节点不批准交易区块,而是批准单独的操作/交易。因此,用户几乎可以立即得到节点150、160、170网络是否批准或拒绝特定请求的操作的确认。

[0097] 参考图4和图5,下面将描述根据本发明的方法的示例。

[0098] 因此,当前类型的交易请求是先前在系统中注册的用户的操作,定义了更改公共分类账的操作。如上所述,这样的操作可以是资金份额的货币转移,或者改变关于用户的信息的操作。注意,操作所针对的所有此类信息通常由每个节点150、160、170存储在由每个节点150、160、170保存的相应同步副本中。

[0099] 本示例中的交易信息包括以下数据。通常,此类数据由客户端110、120、130或用户提供,并由每个涉及的节点150、160、170验证:

[0100] `requesting_user`: 识别请求操作的用户的信息。

[0101] `operation_user`: 操作交易的用户,通常是`requesting_user`。但也可能是级用户因可疑活动而锁定了用户的帐户。

[0102] `counter`: 此用户先前提提交的操作数加一。

[0103] `epoch_ms`: 自UTC 1970-01-01T00:00:00以来的毫秒数。

[0104] `block_id`: `int(epoch_ms/30000)`(注意当前区块直接由当前时间定义)。

[0105] `ref_block_id`: `block_id`减二。

[0106] `user_state`: `operation_user`上所有先前提提交的操作的各个有效负载的散列的XOR。

[0107] `ref_block_state`: 所有操作(注册交易信息)的XOR哈希值,用于在区块上刻下的操作,直到被`reference_block_id`标识的区块为止。

[0108] 注意,在某些类型的交易中,例如转账类型的交易,可能涉及多个用户(例如发送和接收用户)。然而,为了本发明的目的,正是`operation_user`定义了所讨论的交易是针对哪个用户执行的。优选地,每个交易仅针对一个用户执行。

[0109] 在该示例中(见图4),区块大小为30秒,参考区块是当前块`id`减去2。在替代示例中,可以使用其他配置,例如区块大小为1毫秒,`ref_block_id`为`block_id`后面的30000个区块。换句话说,在这样的示例中,`block_id=epoch_ms`和`reference_block_id=block_id-30000`。

[0110] 图5示出了处理所请求交易的更详细方案。交易请求可以发送到任何节点150、160、170。每个节点以两种不同但相似的模式处理传入的交易信息集--它要么接收来自用户的交易请求,要么接收来自另一个节点(例如接收用户请求的节点)的交易信息。

[0111] 首先,针对传入的交易信息,对一些方面进行验证。

[0112] 1、交易信息所定义的操作对有关节点来说是新的,还是该节点已经收到了该交易信息?

[0113] 2、交易计数器、交易时间戳、`user_state`和`ref_block_state`是否都正确(如预期)?

[0114] 3、该操作是否有效并被正确签名,`requesting_user`是否有特权(被授权)进行该操作?

[0115] 4、用户是否已经没有任何待处理的操作了?

[0116] 如果上述1、2、3、4中的任何一项检查失败,那么就会发生以下两种情况之一。如果所讨论的处理节点从另一个节点收到了交易信息,处理节点就会投反对票,而发送节点也会被通知这一投票。另一方面,如果所讨论的处理节点从用户那里收到了交易信息(处理节点是交易请求的接收节点),则交易被快速拒绝,而所讨论的节点没有执行任何更多的工作,交易信息也没有分发到其他节点。

[0117] 在该实施例和其他实施例中,当节点150、160、170在它们之间发送提议的交易时,它们将它们各自的投票附加到通过的信息上。这意味着每个节点150、160、170已经可以从接收到的交易信息中得知该节点是否以其作为第一交易接收节点(从客户端110、120、130接收交易请求)或辅助节点的身份被联系(从另一个节点150、160、170接收交易信息)。优选地,投票信息由发送节点添加到发送的交易信息中,生成的信息由投票节点签名。与用于此签名的私钥相对应的公钥在公共分类账上可用,这就是为什么客户机无法通过加密方式欺骗此类投票。如果接收到交易的第一节点将投“反对”,并且收到的交易信息中没有投票信息,第一节点将快速拒绝而不是要求其他节点考虑该交易。

[0118] 另一方面,如果所有检查1、2、3和4都通过,则处理节点投赞成票,并将有关该投票的信息传播给其他节点,例如通过发送节点。

[0119] 如果交易请求定义了金钱或其他资金的转移,则检查要从中转移资金的相关账户上是否存在足够的资金。如果情况并非如此,则接收节点可以再次快速拒绝该交易。

[0120] 然后,接收节点对交易信息进行签名,并将该用户提交为等待进行中的交易的用户。

[0121] 此后,投票由接收节点收集并保存。如果通过了预定义的赞成票阈值,则提交交易。投票或批准状态由接收节点发送到其他节点。

[0122] 然后所有节点检查交易是否在投票中获得批准。如果不是这种情况,则用户处于未决状态,并且(由至少一个节点,例如接收节点)将交易拒绝发送回用户用来请求交易的客户端110。

[0123] 如果交易获得批准,将发生以下两种情况之一。

[0124] 在第一种情况下,交易请求是“文档更改”类型的操作,换言之,交易信息定义了保存一个或多个包含用户信息的文档的操作。例如,交易信息可以规定“这是我的新电话号码”或“这是我的新公钥”。系统100优选地具有一组明确定义的可能的此类文档,并且可能存在允许的文档类型的列表(见下文)。

[0125] 在操作被注册和/或执行并且因此用户未挂起之后,操作计数器被更新为区块状态和用户状态。然后,发送交易批准,例如发送给请求用户。可以由不同的处理节点发送多个这样的交易批准。

[0126] 如上所述,系统100可以定义多种类型的文档,可能包括以下类型:

| 文件类型               | 说明                                                                              |
|--------------------|---------------------------------------------------------------------------------|
| grid_config        | 描述节点、它们的 URL、它们的公钥和投票计数配置。这决定了节点提交交易所需的投票批准级别。                                  |
| read_auth          | 请求用户在读取请求时提交的 128 位密钥,只有在读取密钥匹配时,节点才会验证和服务这些请求。                                 |
| user_info          | 有关 operation_user 的未经验证的信息。                                                     |
| Verified_user_info | 外部合作伙伴已验证的有关用户的 operation_user 的信息。该信息属于 operation_user,但不能被 operation_user 更新。 |
| write_auth         | 有关 operation_user 的公钥的信息,以及哪些外部用户有权对该用户进行操作(以及操作到什么程度)。                         |
| Wallet             | 包含对价值存储(如数字加密货币钱包或传统帐户)的引用和余额的文档。这是在转移资金交易中操作的。                                 |

[0128] 例如,服务提供公司形式的特定用户可能有权从某个 operation\_user 的钱包中提取价值 2000 股/月的价值量。

[0129] 因此,节点150、160、170中的每一个可以保持这些文件中的每一个,这些文件由处理被请求并通过系统100的所有交易的所有节点150、160、170保持同步。

[0130] 具体地,关于在请求的交易中接收的交易信息中包括的并且包括在节点150、160、170之间流通的交易信息中的用户交易计数器,该参数强制以明确定义的顺序执行关于特定用户的操作。如果每个节点150、160、170对当前计数器的期望值有不同的看法,它就会拒绝操作。例如,如果用户有四个先前注册的交易,并且针对相同的operation\_user并指定counter=7发送新的交易请求,则该交易请求将被拒绝。如果针对同一个用户发送了两个不同的操作,都指定counter=5,则最多其中一个将被投票节点150、160、170中的大多数批准。结果,根据本发明的投票机制将在单个交易的基础上自动处理双重消费拒绝。

[0131] 因此,包含在每个交易信息中的计数器参数对给定用户的操作进行排序,而不对不同用户的操作(彼此相关)排序。事实上,优选的是,系统100中未定义针对不同注册用户执行的交易之间的内部顺序。这实际上提供了更有效的系统100,因为一个和同一块中的所有订单不需要以任何特定的顺序出现。

[0132] 具体关于参数user\_state,该参数也包含在接收到的交易信息中,并在节点150、160、170之间流通。使用此参数,节点多数和用户(关于执行交易的对象)都被迫就用户历史的整个状态达成一致。这使得系统100在密码上不可能欺骗所讨论的用户。即使分类帐的其余部分对用户隐藏,用户也可以验证其历史状态未损坏。

[0133] 具体关于ref\_block\_state参数,它也包含在接收到的交易信息中,并在节点150、160、170之间流通。使用此参数将确保节点多数同意区块状态,返回两步(或与所用参考区块的距离有多大)。这将在节点150、160、170之间没有协商的情况下发生,并在正常投票过程中处理。

[0134] 特别是关于type\_counter,这是一个节点内部参数,它在交易的正常处理期间不分布在节点150、160、170之间,但可以传送到如上所述进行初始化的节点。节点150、160、170各自为每个用户维护type\_counter的历史。特别是,type\_counter是每个用户的映射,表示相关用户为每个计数器请求的操作类型(当相关用户上次请求该类型的交易时,用户交易计数器的值是多少)。这是当前操作计数器为5的特定用户的示例:

[0135] 计数器“ReadAuth”:2

[0136] 计数器“VerifiedUserInfo”:3

[0137] 计数器“WalletHolder”:4

[0138] 计数器“WalletRef”:5

[0139] 计数器“WriteAuth”:1

[0140] last\_epoch\_ms:1548590802702

[0141] operation\_counter:5

[0142] user\_id:“bcPIImBotUSuNKKGox311LA==”

[0143] 例如,在此数据中,我们可以读取用户上次在计数器1处更改其WriteAuth(加密密钥)。此信息可用于执行有效的恢复。

[0144] 使用上面具体描述的一些或全部参数,本发明人已经发现这种类型的系统100可以达到每秒100000个交易的吞吐量,或者甚至更高。这比众所周知的传统数字交易系统(例如使用比特币协议)要多得多。

[0145] 在根据本发明的方法中,在提交交易之前不需要等待协商区块。如果交易在投票程序中获得批准,将立即提交注册并可能进行后续执行。在涉及数字存储的货币存款的货

币交易的示例中,往返交易时间通常将少于一秒,然后接收方可以使用货币。对相关用户的确认速度更快。

[0146] 此外,每个用户只能看到与该用户相关的操作。由于为每个交易更新user\_state参数,用户可以检测并证明系统100是否试图欺骗她,即使她无法看到整个分类帐。

[0147] 由于在每个投票节点150、160、170上对交易的单独批准,并且进一步由于在ref\_block\_state的节点之间没有进行谈判的事实(这只是交易被投票为“赞成”的结果,因此提交),根据本方法的协议也可以以高交易速度运行。使用高性能硬件,1MTPS的吞吐量应该是无法达到的。单独的投票过程还使每个节点150、160、170都具有水平可扩展性。

[0148] 此外,根据本发明的方法具有增加对超级用户能力的支持的潜力,这可能是控制所需要的。例如,在当前类型的交易中交易的资产由账户服务器140保存在封闭系统中的设置的情况下(如上所述),允许控制实例锁定或扣押资产的功能是容易实现的。

[0149] 实施例1:正常处理交易而不拒绝

[0150] 在此示例中,用户向节点#1发送支付请求。下表显示了选择步骤和数据流:

[0151]

| 步骤 | 用户           | 节点#1                               | 节点#2           | 节点#3...       |
|----|--------------|------------------------------------|----------------|---------------|
| 1  | 向节点#1 发送支付请求 |                                    |                |               |
| 2  |              | 收到请求,<br>投“赞成”                     |                |               |
| 3  |              | 将 请 求 和<br>“赞成”投票<br>转发给所有<br>其他节点 |                |               |
| 4  |              |                                    | 以“赞成”投票<br>票回应 | 以“赞成”投票回<br>应 |
| 5  |              | 收集多数票<br>(“赞成”)                    |                |               |
| 6  |              | 执行交易                               |                |               |
| 7  |              | 将投票收集<br>发送到其他<br>节点               |                |               |
| 8  |              | 通过频道向                              | 接收多数票          | 接收多数票并履       |

|        |    |      |               |                    |                    |
|--------|----|------|---------------|--------------------|--------------------|
| [0152] |    |      | 付款人和收款人用户发送投票 | 并履行交易              | 行交易                |
|        | 9  |      |               | 通过频道向付款人和收款人用户发送投票 | 通过频道向付款人和收款人用户发送投票 |
|        | 10 | 接收投票 |               |                    |                    |

[0153] 实施例2:转账操作

[0154] 在该示例中,请求将价值100的金额从第一钱包“1”转移到第二钱包“2”。

[0155] 交易请求可能如下所示(伪代码):

```
{
  "payload": {
    "block_id":28,
    "counter": 4,
    "data": {
      "Amount": "100",
      "from_wallet_id ":"1",
      "to_wallet_id ":"2"
    },
    "epoch_ms": 1549638968221,
    "grid_config_id": 1,
    "operation_type": "TRANSFER"
    "ref_block_id":26,
    "ref_block_state": "11111111...",
  }
```

```

        "user_state": "SHA-256:10101010...",
    },
    "signature": {
        "key_id": "1234",
[0157]    "payload_hash": "SHA-256:11110000...",
        "proof": "00010001..."
    },
    "user_id": "test"
}

```

[0158] 当接收节点150接收到包含该交易信息的交易请求时,执行以下验证:

[0159] 1、用户“test”的当前用户计数器为“3”。

[0160] 2、用户“test”没有挂起的操作。

[0161] 3、节点当前时间与epoch\_ms相差不远。

[0162] 4、钱包“1”属于用户“test”。

[0163] 5、钱包“1”的余额至少为100。

[0164] 6、给定的epoch\_ms确实属于block\_id为“28”的区块。

[0165] 7、将block\_id引用到区块“28”实际上是区块“26”。

[0166] 8、区块“26”的区块状态实际上是“11111111...”。

[0167] 9、用户“test”的user\_state实际上是“10101010...”。

[0168] 10、有效负载哈希实际上是“11110000...”。

[0169] 11、证明“0001001”是用户“test”的有效签名。

[0170] 当节点150、160、170中的任何一个收集到显示对所请求交易的超过阈值批准的总投票数时,该收集节点按以下步骤执行交易处理:

[0171] 1、将金额100从钱包“1”移动到钱包“2”。

[0172] 2、将用户“test”的计数器从“3”更新为“4”。

[0173] 3、将用户状态设置为先前用户状态和payload\_hash的XOR。

[0174] 4、将block\_id“28”的block\_state设置为区块“28”的先前block\_state和payload\_hash的XOR。

[0175] 因此,对于成功的交易,这些步骤由网络中的每个节点150、160、170使用当时所讨论节点的相应本地信息来执行。然而,这些步骤可以由不同的节点在不同的时间点执行,这取决于网络中的信息流。

[0176] 在注册交易的完整处理之前和之后,每个节点150、160、170中的状态如下表:

| 对象               | 交易前         | 交易后         |
|------------------|-------------|-------------|
| block_state (26) | 11111111... | 11111111... |
| block_state (28) | 01010101... | 10100101... |
| user_state       | 10101010... | 01011010... |

|              |     |     |
|--------------|-----|-----|
| user_counter | 3   | 4   |
| 余额钱包“1”      | 100 | 0   |
| 余额钱包“2”      | 100 | 200 |

[0178] 实施例3:管理员写入用户信息

[0179] 在该示例中,管理员用户(它是一个半超级用户,有权对其他用户执行某些操作)更新存储在上述同步文件中的用户信息。

[0180] 交易请求可能如下所示(伪代码):

```
{  
  "payload":  
  {  
[0181]    "block_id": 51666252,  
    "counter": 29,  
    "data": {  
      "date_of_birth": null,
```

```

        "email": null,
        "full_name": "John Doe",
        "mobile": "+15555555555"
        "user_id": "EZ2j79j2QHus84LnSYI8GA==",
    },
    "epoch_ms": 1549987564172,
    "grid_config_id": 1,
    "hash_scheme": "SHA-256",
    "operation_type": "UserInfo",
    "ref_block_id": 51666250,
    "ref_block_state": "hjYG74FDkokuytju76gtFR==",
    "user_state":
    "SHA-256:qjfsPPN17Sae/YWWmgyfd/DVWuzn7TyNeh4GZY2YgvE="
[0182] },
    "poa": "admin",
    "signature":
    {
        "proof":
        "448K78Ur5P5dcciPDjJheq+Us8YDb3owArR18po9rYX2sWQGQvoqhcZ9
        sih6Qy28wGR1qw3f4TuKG0A0qKPdljwfAmnyoK5G/Hgza5IvY5NtE/MBvUu
        WI5AAYmQq5hAx",
        "payload_hash":
        "SHA-256:plil42vQ7ap2BgoYPRcGrmAPIbJISg3KRsrqPjMIBK0="
    }
    "user_id": "EZ2j79j2QHus84LnSYI8GA==",
}

```

[0183] 当节点150、160、170验证操作并通过投票批准后,每个节点将通过执行以下步骤来执行操作(注册/执行交易):

[0184] 1、将data子句下的信息写入属于用户“EZ2j...”的user\_info文档中。

[0185] 2、将用户计数器从“28”更新为“29”。

[0186] 3、以与示例2(以上)相同的方式更新user\_state,但在此操作中使用payload\_hash。

[0187] 4、区块id的处理方式与示例2相同。

[0188] 实施例4:双重消费交易

[0189] 在该示例中,用户试图通过向所有节点发送具有相同计数器的两个支付指令来欺骗系统100,希望这两个支付都通过。

[0190] 下表显示了选择步骤和数据流:

[0191]

| 步骤 | 用户                        | 节点#1                       | 节点#2              | 节点#3...                |
|----|---------------------------|----------------------------|-------------------|------------------------|
| 1  | 向所有节点<br>发送双重支付指令 p1 和 p2 |                            |                   |                        |
| 2  |                           | 接收 p1 并投赞成票                | 接收 p1 并投赞成票       | 接收 p2 并投赞成票            |
| 3  |                           | 接收 P2 并快速拒绝<br>(原因: p1 未决) | 接收 P2 并快速拒绝       | 接收 P1 并快速拒绝            |
| 4  |                           | 从节点 3 接收 p2, 投反对票          | 从节点 3 接收 p2, 投反对票 | 从节点 1/节点 2 接收 p1, 投反对票 |
| 5  |                           | 收集 p1 多数                   | 收集 p1 多数          | 收集 p2 多数票,             |

[0192]

|    |      |                  |                  |               |
|----|------|------------------|------------------|---------------|
|    |      | 票，赞成             | 票，赞成             | 反对            |
| 6  |      | 执行 p1            | 执行 p1            | 将 p2 保存为已拒绝   |
| 7  |      | 将投票收集发送到其他节点     | 将投票收集发送到其他节点     | 将投票收集发送到其他节点  |
| 8  |      | 接收收集，将 p2 保存为已拒绝 | 接收收集，将 p2 保存为已拒绝 | 接受多数票并执行动作 p1 |
| 9  |      | 通过频道发送投票         | 通过频道发送投票         | 通过频道发送投票      |
| 10 | 接收投票 |                  |                  |               |

[0193] 实施例5:被黑客攻击的节点,双重消费,电网瘫痪

[0194] 这里,用户尝试通过向所有节点150、160、170发送具有相同计数器的两个支付指令来欺骗系统100,希望这两个支付都通过。这次用户得到了节点2的帮助,节点2已被黑客入侵以批准所有请求。因此,在这种情况下,节点2没有按照本发明应有的工作,而是被非法修改。

[0195] 下表显示了选择步骤和数据流:

[0196]

| 步骤 | 用户                  | 节点#1     | 节点#2 (被黑客攻击) | 节点#3...   |
|----|---------------------|----------|--------------|-----------|
| 1  | 向所有节点发送支付指令 p1 和 p2 |          |              |           |
| 2  |                     | 接收 p1 并投 | 接收 p1 并投     | 接收 p2 并投赞 |

[0197]

|   |  | 赞成票                     | 赞成票               | 成票                         |
|---|--|-------------------------|-------------------|----------------------------|
| 3 |  | 接收 P2 并快速拒绝             | 接收 p2 并投赞成票       | 接收 P1 并快速拒绝                |
| 4 |  | 从节点 2/节点 3 接收 p2, 投反对票  | 从节点 3 接收 p2, 投反对票 | 从节点 1/节点 2 接收 p1, 投反对票     |
| 5 |  | 收集 p1 多数票, 赞成           | 收集 p1、p2 多数票, 赞成  | 收集 p2 多数票, 赞成              |
| 6 |  | 执行 p1                   | 执行 p1             | 执行 p2                      |
| 7 |  | 将投票收集发送到其他节点            | 将投票收集发送到其他节点      | 将投票收集发送到其他节点               |
| 8 |  | 准备执行 p2 计数器已使用。注意系统已损坏。 | 接收收集, 将 p2 保存为已拒绝 | 接收 p1 的多数票。计数器已使用。注意系统已损坏。 |
| 9 |  | 拒绝所有交易直到手动解决            | 被黑客攻击的节点将被手动停止和恢复 | 拒绝所有交易直到手动解决               |

[0198] 实施例6:外部提款、私人支付用例

[0199] 该示例示意性地说明了系统100可以如何提供私人支付(系统100已知但收款人未知的付款人身份)。在这里,外部用户(例如在线新闻媒体)已被授权以有限的权力代表未知用户提取资金,例如为访问个别新闻文章而付款。

| [0200] | 步骤 | 用户                      | 外部用户                   | 节点                                                |
|--------|----|-------------------------|------------------------|---------------------------------------------------|
|        | 1  | 授权外部用户使用上限（最高 100 美元/月） |                        |                                                   |
|        | 2  |                         |                        | 为用户保存外部身份验证                                       |
|        | 3  | 为其钱包添加外部私人参考            |                        |                                                   |
|        | 4  |                         |                        | 保存私人钱包参考                                          |
|        | 5  | 请求阅读文章，只发送私人参考          |                        |                                                   |
|        | 6  |                         | 执行付款，从私人参考到报纸钱包，0.1 美元 |                                                   |
|        | 7  |                         |                        | 由于未违反每月 100 美元的上限，执行付款。系统知道私人参考的真正所有者是谁。通过频道发送投票。 |
| [0201] | 8  | 接收有关付款的完整信息             | 接收已从私人参考收到资金的信息        |                                                   |
|        | 9  |                         | 用文章服务私人客户              |                                                   |

[0202] 实施例7:分类账的“前沿状态”

[0203] 每个节点150、160、170被配置为(并且至少一些节点实际上将)保存系统100的交易历史中的每个交易,以便在遇到挑战时重新创建整个历史并证明分类账状态。然而,为了

能够作为系统100中的功能节点运行,并批准/拒绝个别交易,只需要该状态的一小部分。

[0204] 例如,假设用户先前已将write\_auth文档中包含的她的公钥更新了100次(在各自先前注册的交易中)。当来自同一用户的新交易请求发布时,在查看此交易签名的验证时,只有当前(最后)版本的write\_auth相关。

[0205] 特定钱包的情况类似。如果钱包的余额是1000,那么导致这个精确余额的操作顺序并不重要,当用户想要转账100.0时,重要的是钱包中有足够的资金来支持这样的交易。

[0206] 因此,每个单独的节点150、160、170只需要以下信息以便从特定时间点恢复操作,并继续处理交易信息:

[0207] 1、分类帐中每个文档的最新版本

[0208] 2、每个钱包的余额

[0209] 3、系统区块大小配置中的最后两个区块状态或最后一个区块状态的数量(但是至少有两个最后区块状态)

[0210] 上述第1-3点中的信息称为系统100的“前沿状态”。需要注意的是,“前沿状态”信息明显少于整个分类账。即整个账本大小与操作次数成正比,大致为 $O(\text{time} \times \# \text{users})$ ,而前沿状态仅大致为 $O(\# \text{users})$ 。

[0211] 然后假设节点160已经停机很长时间,并且其余节点150、170在该停机时间内已经批准并执行了大量交易。恢复节点160的一种方式是从节点160宕机之前的最后已知状态恢复交易。如果按照这个策略,要赶上节点160需要很长时间。相反,可以手动停止另一个节点170,从而不响应任何传入交易。然后,该另一个节点170可以保存它的前沿状态并将这个信息发送到未更新的节点150。在该操作之后,这两个节点160、170都可以从该状态恢复操作。根据系统100的配置,从该状态开始,节点160、170可能仅比其他节点150落后大约1-2分钟,因此快速追赶是可行的。

[0212] 此外,所有节点150、160、170都被配置为(通过上面讨论的数字接口)就特定区块中发生的所有交易相互询问。因此,如果一个特定节点只落后几个区块,它可以向另一个完全最新的节点询问这些区块中的交易。然后,所讨论的追赶节点可以开始在其分类帐上注册这些操作并根据情况执行文档更新。只要追赶节点比传入交易快,它就会相对快速地更新。

[0213] 通过使用上述前沿状态功能和后面的区块操作请求,每个节点150、160、170能够在各种长度的停机时间之后快速赶上。在恢复已进入损坏状态的被黑客攻击的节点时,也可以使用相同的机制(如上面的示例5)。使用此示例5,手动恢复过程可能如下所示:

[0214] 1、停止所有节点上的所有操作。

[0215] 2、无视被黑客攻击的节点。

[0216] 3、递归删除获得多数票批准但任何非黑客攻击节点拒绝执行的操作(在示例5的情况下,有两个这样的操作,即在同一个操作计数器上操作的两个双重消费)。

[0217] 4、验证所有未受黑客攻击的节点的block\_state是否相同。

[0218] 5、擦干净被黑客攻击的节点。

[0219] 6、保存其中一个非被黑客攻击节点的前沿状态并将其放入被黑客攻击节点的数据库中。

[0220] 7、恢复操作。

[0221] 8、联系来自两个双重消费操作的收款人,这些操作将获得有效付款的加密收据,而无需将这些金额记入此分类账的恢复版本中各自的帐户余额中。

[0222] 9、协商对这些用户的潜在补偿,这取决于他们由于(在这种情况下)付款证明损坏而发送的价值。

[0223] 此类过程可由控制实体或节点(图1中未示出)执行,其具有与所有其他节点相关的超级用户状态。

[0224] 可用于本发明的一些重要方面包括以下。

[0225] 首先,用户操作计数器包含在发送到系统100的每个交易请求中。

[0226] 其次,用户状态包含在发送到系统100的每个交易请求中。

[0227] 第三,系统(区块)状态可以包含在发送到系统100的每个交易请求中。

[0228] 第四,每个用户都有一个单独的区块链(如上所述),其中每个区块可能只包含一个操作/交易。

[0229] 第五,系统100可以保存类型计数器的当前和历史分类帐。

[0230] 第六,任何节点都可以在离线一段时间后上线并产生与自身一致的前沿状态,然后根据该前沿状态信息恢复运行。前沿状态比整个状态小得多,可以被其他节点用来恢复操作。

[0231] 以上,已经描述了优选实施例。然而,对于技术人员显而易见的是,在不脱离本发明的基本思想的情况下可以对所公开的实施例进行许多修改。

[0232] 例如,除了在此描述的那些特征之外的许多其他特征也可以在配置为执行根据本发明的方法的系统100中实现。

[0233] 一般而言,本文提供的不同示例的各个方面是可自由混合的。选择这些示例中的每一个都是为了突出本发明的某些重要方面。

[0234] 因此,本发明不限于所描述的实施例,而是可以在所附权利要求的范围内变化。

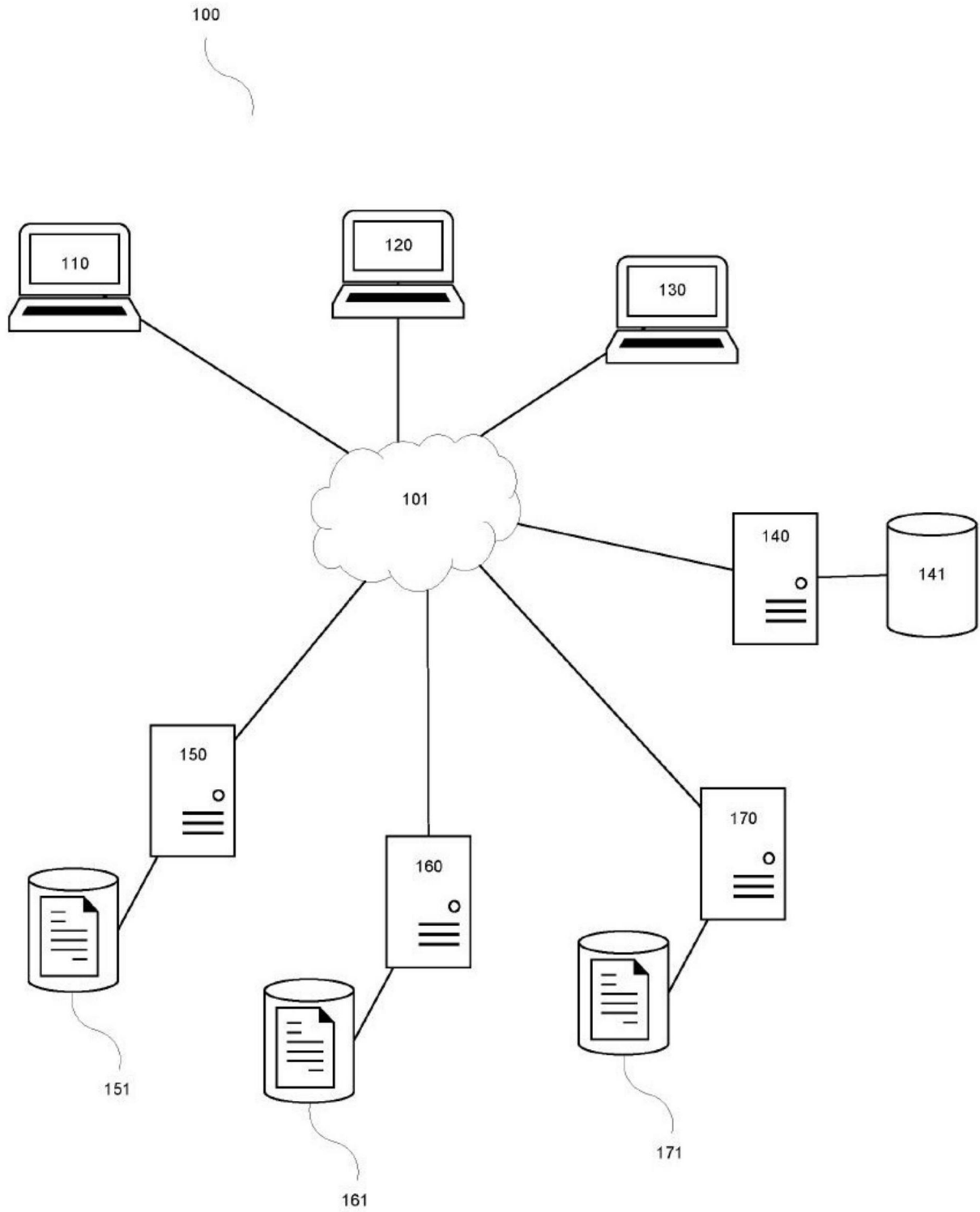


图1



图2

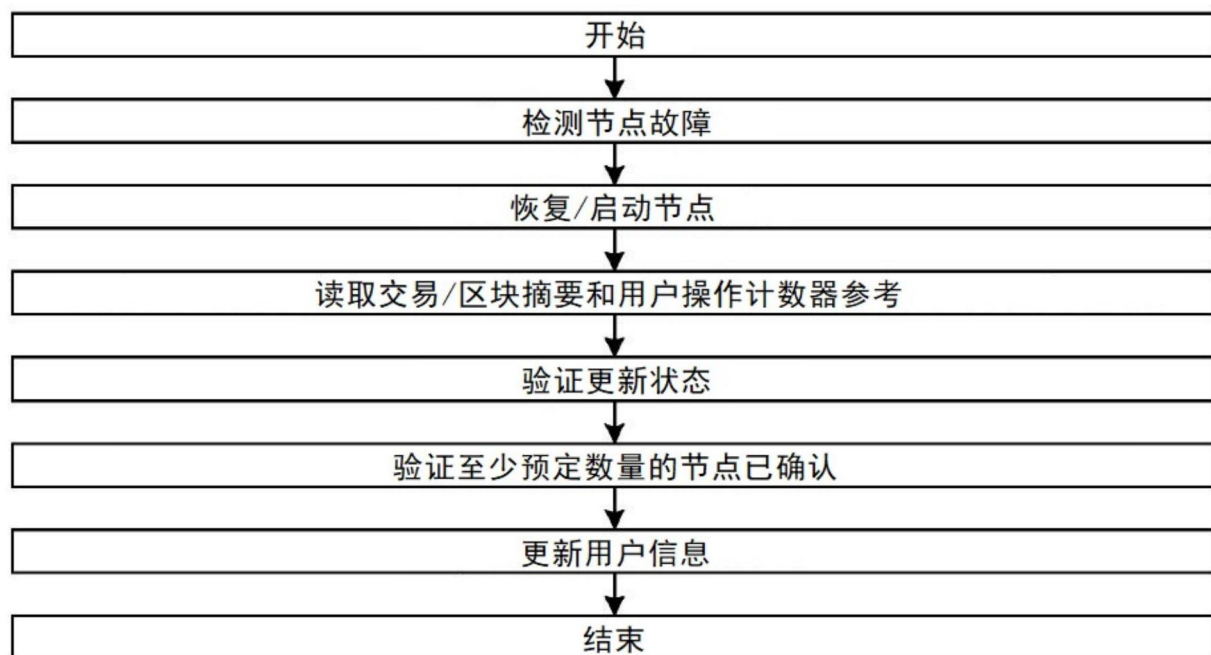


图3

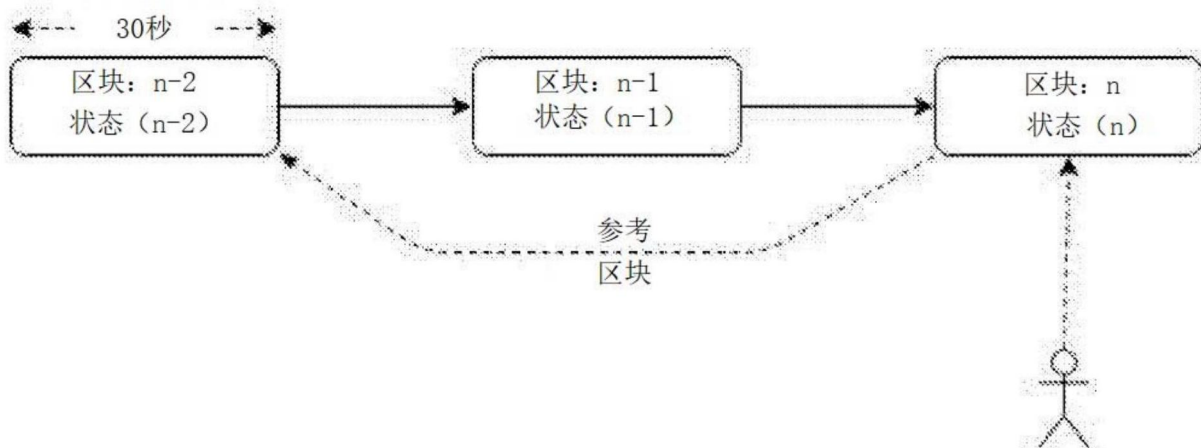


图4

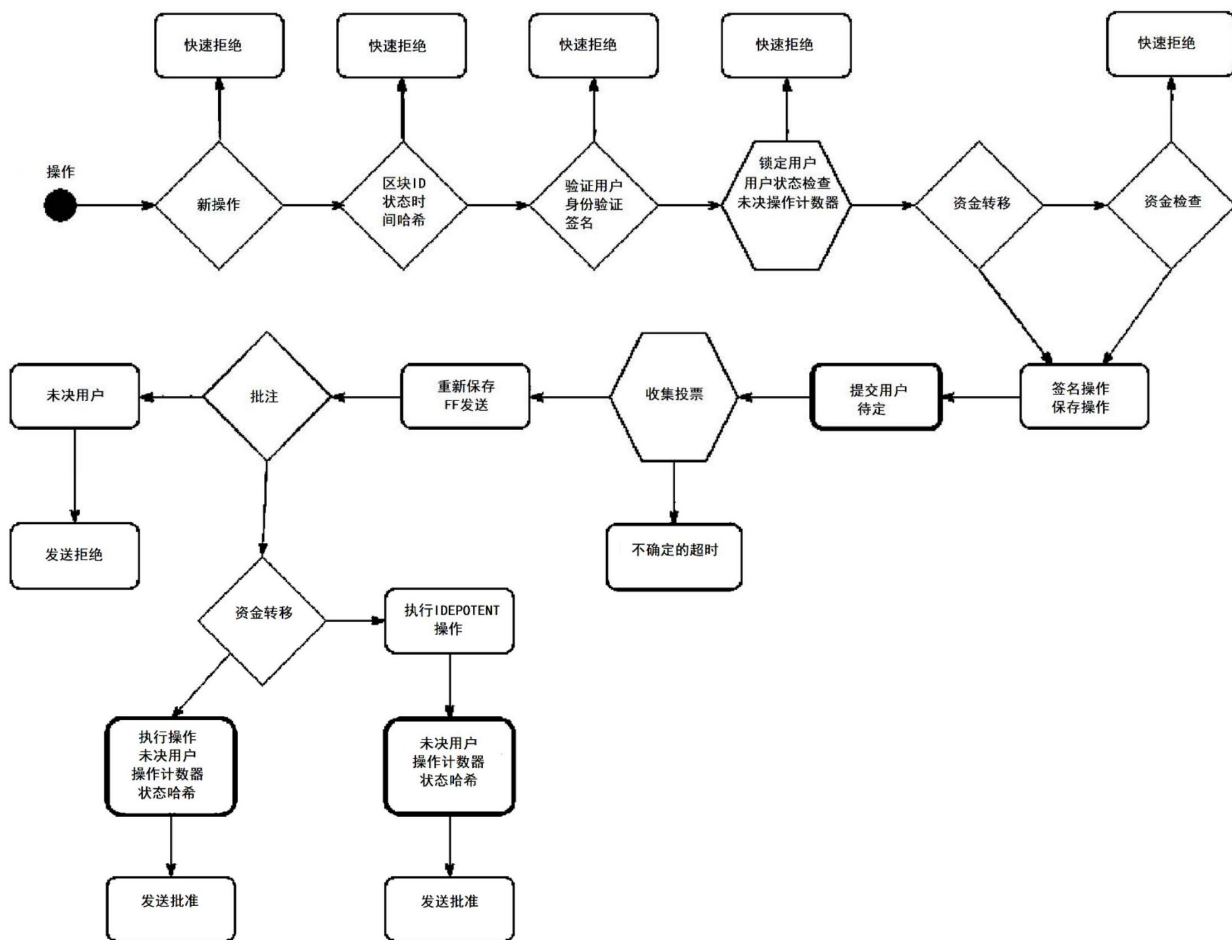


图5