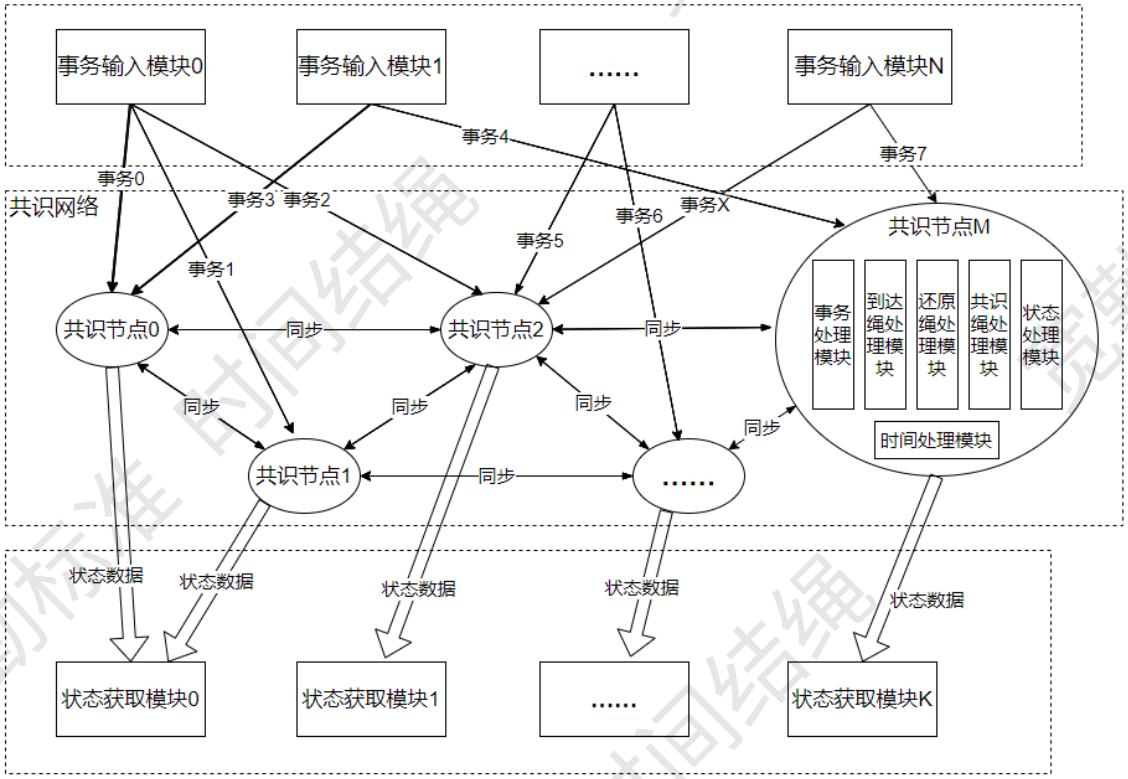
说　明　书　摘　要

本申请公开了一种事务定序共识方法和系统，其中，所述方法包括：共识节点不间断地构造连续的前向依赖数据区块，将接收到的事务立即引入当前数据区块中，形成事务到达本共识节点相对时序的证据；共识节点之间同步所述证据，各自还原出对于事务进入网络的时序判断，并经共识处理形成事务时序共识；各共识节点处理共识事务完成对系统状态的一致性变更；进一步地，定义了一种共识时间。通过本申请方案，解决现有技术中因信息损失而造成的事务定序能力不足，因竞选主节点而造成不公平竞争和浪费，因主节点主观操纵而造成的事务定序失公平，以及由上述问题衍生的问题。

摘　　要　　附　　图



权　利　要　求　书

1．一种事务定序共识方法，其特征在于，所述方法包括：

获取待共识事务和相应的事务信息，所述事务信息包括事务编号、事务输入模块信息、事务入口节点信息，事务待执行信息和获取待共识事务的路径信息；

根据所述获取待共识事务的路径信息，将所述待共识事务划分为本地事务和外地事务；

验证、解析、同步所述待共识事务；

获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳；

同步不同共识节点之间的本地到达绳，得到同步后各节点到达序信息；

根据预设还原规则分析所述各节点到达序信息，得到本节点中两两事务的还原序以及一组事务连续的还原序信息，构建本地还原绳；

交换和同步不同共识节点之间的本地还原绳，得到同步后各节点还原序信息；

根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，所述共识绳包含已共识事务；

根据对所述共识绳解析，得到共识绳中已共识事务；

根据已共识事务更新状态数据。

2．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，更新状态数据之后，所述方法还包括：

共识节点根据共识绳的状态变化轨迹得到共识时间。

3．根据权利要求 2 所述的一种事务定序共识方法，其特征在于，所述共识时间的当前时间包括当前时刻值和所述当前时刻值的校验值；

当前时刻值为所述共识序中当前已共识事务的个数；

获取所述共识时间的当前时间为时间戳。

4．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，在获取待共识事务和相应的事务信息之前，还包括：

获取事务触发信息；

获取共识网络中共识节点信息；

根据所述事务触发信息和共识节点信息，确定待共识事务的事务入口节点信息。

5．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，所述事务入口节点信息通过预设分配规则为所述待共识事务分配的入口节点，其中，所述预设分配规则包括指定、轮询、随机、就近和负载均衡；

根据所述事务入口节点信息区分所述待共识事务。

6．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，验证、解析所述待共识事务，包括：

根据预设验证项对所述待共识事务进行验证，得到验证结果；

当验证结果通过时，将所述待共识事务标记为已验证待共识事务；

根据预设解析规则对所述已验证待共识事务进行解析，得到解析结果；

将所述已验证待共识事务标记为已解析待共识事务。

7．根据权利要求 6 所述的一种事务定序共识方法，其特征在于，同步所述待共识事务，包括：

获取所述已验证待共识事务及当前共识时间戳；

创建待共识事务同步消息，所述待共识事务同步消息包括待同步事务、消息发布时间戳、共识节点签名；

根据网络拓扑数据，按预设生成原则生成已验证待共识事务同步算法；

根据所述已验证待共识事务同步算法对所述已验证待共识事务进行同步，得到已同步待共识事务。

8．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳；同步不同共识节点之间的本地到达绳，包括：

初始化共识节点的本地到达绳；

不间断地创建绳结且延长所述本地到达绳，所述绳结包括绳结编号、引入绳结数据和校验项；

当所述共识节点接收到本地事务时，将所述本地事务引入所述本地到达绳的当前绳结；

当在满足预设的本地到达绳的同步触发条件时，向本节点外共识节点同步所述本地到达绳中未同步部分；

根据预设到达规则接收、验证和同步外地到达绳；

解析所述本地到达绳和所述外地到达绳，得到同步后各节点到达序信息。

9．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，所述绳结编号为连续得自然数，且于所述绳结的产生时间相对应；

将所述本地到达绳作为本节点的计时手段；

所述本节点计时手段的当前时刻值为所述本地到达绳的当前的绳结编号。

10．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，验证外地到达绳的方法包括：

获取待验证外地到达绳；

根据预设到达绳验证项对所述待验证的外地到达绳段进行验证，得到验证结果；其中，所述预设到达绳验证项包括外地到达绳的签名、外地到达绳中每个绳结、每个绳结对应的事务、每个绳结的校验值；

对所述待验证的外地到达绳段进行验证包括随机验证、并行验证和协作验证。

11．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，同步外地到达绳包括：

获取所述外地到达绳；

创建外地到达绳同步消息，所述外地到达绳同步消息包括待同步外地到达绳、消息发布时间戳、共识节点签名；

将所述外地到达绳写入所述外地到达绳同步消息，生成外地到达绳同步算法；

根据所述外地到达绳同步算法向其他共识节点发送外地到达绳同步消息，得到已同步外地到达绳。

12．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器和调度器，包括：

依次异步调用空闲的打结器进行打结操作；

生成绳结编号、确定前序绳结以及待映入绳结的事务，并创建引入绳结。

13．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器，且 N 个打结器各自有唯一的编号并持续竞争一个互斥锁，所述互斥锁携带一个整数变量 a 用于生成绳结编号和一个变量 b 用于记录最近一次竞争到互斥锁的打结器的编号；包括：

当一个打结器竞争到互斥锁后，将互斥锁所携带的整数变量 a 加 1 从而生成当前绳结编

号；

读取互斥锁变量 b 从而获取上一个竞争到互斥锁的打结器的编号；

将变量 b 置为本打结器的编号，并判定是否有待引入绳结的事务；

当有待引入绳结的事务，则在所述本地到达绳上打事务结；

当没有待引入绳结的事务，则立即释放互斥锁并打空结；

其中，所述事务结包括本打结器的编号、上一个竞争到互斥锁的打结器的编号；所述空结为没有引入事务且已创建完毕的绳结。

14．根据权利要求 12 或 13 所述的一种事务定序共识方法，其特征在于，所述本地到达绳的打结包括：

获取本地到达绳中含有当前结的绳段；

统计所述含有当前结的绳段中事务结的个数占所述含有当前结的绳段中绳结总数的占比；

当所述占比超过预设阈值时，增加本地到达绳的打结器；

当所述占比低于预设阈值时，减少本地到达绳的打结器。

15．根据权利要求 8 所述的一种事务定序共识方法，其特征在于，同步不同共识节点之间的本地到达绳之前，包括：

对所述本地到达绳进行分段处理，得到到达绳段；

根据预设压缩规则对所述到达绳段的进行压缩处理；

所述压缩处理保留所述到达绳段上的事务结，并是删除所述到达绳段的非标记用空结。

16．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，两两事务的还原序以及一组事务连续的还原序信息，包括：

获取所述到达序信息中涉及的事务信息；

在所述到达序信息中涉及的事务中选择两个事务；

判断所述两个事务的还原序，输出所述两个事务的还原序信息和对所述两个事务的还原序信息的置信度；

迭代至获得所述到达序信息中一组事务连续的还原序。

17．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，交换不同共识节点之间的本地还原绳，包括：

获取共识节点信息，获取到达绳、还原绳和共识绳的作为初始值；

根据所述初始值生成随机数；

根据共识节点信息和所述随机数产生交换还原序信息的节点组合。

18．根据权利要求 1 所述的一种事务定序共识方法，其特征在于，根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，包括：

获取本节点还原绳中未共识且已稳定的还原绳段；

接收其他 N 个共识节点的对应的请求未共识且已稳定的还原绳段；

根据已获取的到达序信息和还原序信息检查外地还原绳；

将接收的请求未共识且已稳定的还原绳段与本节点对应的共识且已稳定的还原绳段比较，保留不同绳段中相同率大于三分之二的时序关系；

重复与多个请求未共识已稳定的还原绳段比较，直到获得相同的时序关系，得到共识序信息，构建共识绳。

19 根据权利要求 1 所述的一种事务定序共识方法，其特征在于，根据已共识事务更新状态数据，包括：

按照共识序信息解析共识绳中已共识事务；

根据所述已共识事务创建事务执行队列；

将所述已共识事务写入所述事务执行队列；

按队列处理规则，通过执行器对所述已共识事务执行队列中的事务进行处理。

20．根据权利要求 19 所述的一种事务定序共识方法，其特征在于，执行器对所述执行队列中的事务进行处理，包括：

执行器获取所述已共识事务的待执行内容；

检查所述待执行内容；

将所述待执行内容转换为可执行指令；

当对应事务满足可执行指令的执行条件，按照可执行指令变更状态数据。

21．一种事务定序共识系统，其特征在于，所述系统包括：

事务输入模块，用于产生待共识事务；

共识网络，用于对所述待共识事务分析处理，获得所述待共识事务的共识序；

状态获取模块，用于呈现状态数据的状态变化。

22. 根据权利要求 21 所述的一种事务定序共识系统，其特征在于，所述事务输入模块包括：

事务发起子模块，用于获取事务触发信息，

事务入口节点分配子模块，从共识网络中获取共识节点信息，根据预设入口节点分配规则、待共识事务和所述共识节点，确定所述待共识事务的入口节点；

事务发送子模块，用于将待共识事务传递给入口节点。

23. 根据权利要求 21 所述的一种事务定序共识系统，其特征在于，所述共识网络包括 N 个共识节点，所述共识节点包括：

事务处理子模块，用于处理本地事务和外地事务；

到达绳处理子模块，用于处理处理本地到达绳和外地到达绳数据；

还原绳处理子模块，用于处理本地还原绳和外地还原绳数据；

共识绳处理子模块，用于处理共识绳数据；

状态处理子模块，用于处理状态数据，并于所述状态获取模块进行数据交互；

时间处理子模块，用户获取时间戳，且验证所述时间戳。

24. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述事务处理子模块包括本地事务处理单元、外地事务处理单元、事务标记单元、事务存储单元；

所述本地事务处理单元包括本地事务获取子单元、本地事务验证子单元、本地事务解析子单元、本地事务同步子单元；

所述外地事务处理单元包括外地事务获取子单元、外地事务验证子单元、外地事务解析子单元、外地事务同步子单元；

所述事务标记单元，用于标记事务状态；

所述事务存储单元，用于存储本地事务、外地事务信息。

25. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述到达绳处理子模块包括本地到达绳处理单元、外地到达绳处理单元、到达序信息存储单元、到达序信息分析单元、到达绳处理优化单元；

所述本地到达绳处理单元包括本地到达绳初始化子单元、本地到达绳打结子单元、本地到达绳同步子单元、本地到达绳存储子单元；

所述外地到达绳处理单元包括外地到达绳获取子单元、外地到达绳拼接子单元、外地到达绳验证子单元、外地到达绳解析子单元、外地到达绳同步子单元、外地到达绳存储子单元；

所述到达序信息存储单元，用于存储本地和外地的到达序信息；

所述到达序信息分析单元，用于分析到达序信息；

所述到达绳处理优化单元，用于响应外部输入并优化到达绳的处理。

26. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述还原绳处理子模块包括到达序信息获取单元、本地还原绳生成单元、还原绳同步单元、本地还原序信息存储单元、外地还原绳同步单元，外地还原绳解析单元，外地还原绳验证单元，外地还原序信息存储单元、还原绳共识序生成单元。

27. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述共识绳处理子模块包括共识绳初始化单元、共识绳打结单元、共识绳事务验证单元、共识绳事务解析单元、共识绳事务执行单元、共识绳数据分析单元、共识绳同步单元、共识绳验证单元、共识绳状态输出单元。

28. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述状态处理子模块包括状态数据初始化单元、状态数据更新单元、状态数据验证单元、状态数据服务单元、状态数据存储单元、状态数据同步单元；

其中，所述状态数据服务单元包括状态数据订阅管理子单元、状态数据主动推送子单元、状态数据请求响应子单元、状态数据权限管理子单元。

29. 根据权利要求 23 所述的一种事务定序共识系统，其特征在于，所述时间处理子模块包括时间定义单元、时间解释单元、时间验证单元、时间标记单元、时间获取单元、时间同步单元。

30. 根据权利要求 21 所述的一种事务定序共识系统，其特征在于，所述状态获取模块包括状态呈现子模块、状态请求子模块、状态订阅子模块、状态合成子模块、状态存储子模块、状态分析子模块。

31. 一种共识节点设备，应用于权利要求 11-20 任意一项所述一种事务定序共识系统，其特征在于，所述共识节点设备包括入口节点设备、到达序节点设备、还原序节点设备、共识序节点设备、状态节点设备、时间节点设备；

所述入口节点设备与所述到达序节点设备、所述状态节点设备数据连接；所述到达序节点设备与所述还原序节点设备数据连接；所述还原序节点设备与所述共识序节点设备数据连接；所述共识序节点设备与所述状态节点设备数据连接；

所述时间节点设备与所述入口节点设备、所述到达序节点设备、所述还原序节点设备、所述共识序节点设备、所述状态节点设备数据连接；

全节点设备具有入口节点、到达序节点、还原序节点、共识序节点、状态节点和时间节点的功能。

32. 一种共识应用程序，应用于权利要求 11-20 任意一项所述一种事务定序共识系统，其特征在于，所述共识应用程序包括多个事务输入模块、多个状态获取模块和应用部署模块；

所述应用部署模块，用于管理与共识应用相关的事务执行器和状态数据在共识节点的部署。

说　　明　　书

一种事务定序共识方法和系统

技术领域

本申请涉及计算机技术领域，尤其涉及一种事务定序共识方法和系统。

背景技术

事务定序共识是指对一组事务进行排序并就所述排序达成共识，是区块链技术的核心内容。事务定序共识是区块链技术的核心目的。区块链系统的本质是去中心化的网络时钟，各节点对时间的解释和推进规则有统一的理解，但时间推进权不由单一节点垄断，也无法预测由哪个节点完成下一次时间推进。每次时间推进都依赖上一个时间戳和可能的一组已接收但未绑定时间戳的事务而产生一个新的时间戳。节点对时间戳的依赖关系和事务与时间戳的绑定关系达成共识，并以此确定事务的因果顺序，即，与同一个时间戳绑定的事务为并发，并晚于上一个时间戳所绑定的事务，以此类推所有时间戳的事务。

区块链就是一种存储时间戳的依赖关系和事务与时间戳绑定关系的数据结构，而区块链技术的共识和密码学方法使区块链实现一定程度上不可逆且不可篡改的增量存储。

现有，区块链技术的事务定序共识方法一般包括“选主、出块、验证、上链”四个步骤。“选主”就是共识网络中各个节点就成为拥有时间推进权的主节点进行竞争或选举；“出块”是指主节点将自己产生的时间戳和一组预先选定的绑定该时间戳的事务打包成区块并传播给其他节点；“验证”和“上链”指的是共识节点验证主节点所出区块，将通过验证的区块写入区块链并执行。选主和出块的本质是选出一个主节点来完成一次时间推进。

然而，节点在接收事务时没有可举证的计时手段，而是把事务无差别地接收进缓冲区，也就丢失了事务到达本节点的时序信息，导致节点在定序时缺乏时序性的客观依据，而不得不利用手续费等非时序性的信息依据，也就给了事务以高额手续费获取定序优先权的机会。主节点时间推进权，包括选择待绑定事务、产生时间戳、决定因果、获得奖励等特权，选主过程可能会引发各节点非理性的竞争，耗费大量资源。节点在选择事务打包区块时存在主观性，往往依据的是使主观利益最大化而不是客观的事务时序证据，节点也无从获得客观证据，这会造成事务定序结果失真，本质上是对事务真实时序的篡改行为。

现有区块链局限于一定程度解决非竞争性事务的存证问题和双花问题，但竞争性事务对事务定序的客观公平性提出更高要求，以少数节点主观定序并强制共识为特征的现有技术不能满足要求，原因在于，第一，现有技术中节点缺乏可举证的计时手段导致在定序时缺乏信息支持，第二，借鉴狭义相对论的现有分布式定序基础理论不适用于绝对时间下的定序共识场景。以时间和信息视角发明新技术是一种趋势。

发明内容

本申请提供了一种事务定序共识方法和系统，用于解决现有技术中因信息损失而造成的事务定序能力不足，因竞选主节点而造成不公平竞争和浪费，因主节点主观操纵而造成的事务定序失公平，以及由上述问题衍生的问题。

为表述方便，事务进入共识网络的真实时序称作“本真序”，事务到达某节点的时序称作所述节点的“到达序”，节点对事务进入共识网络的时序的判断称作“还原序”，节点对事务进入网络的时序的共识称作“共识序”。

从信息的视角，本公开内容的发明构思在于：由于事务进入网络的时序是客观形成的，通过一系列计算和通信手段，节点捕捉和收集事务与共识网络结合过程中产生的各碎片原始信息，结合已共识的推理方法，从所收集的信息中尽量复原出事务进入共识网络的时序信息并达成共识，使事务能按照接近客观形成的时序被定序和执行，避免节点主观操纵事务定序，并且由于原始信息已经消除了大量不确定性，各节点更容易达成共识。

从时间的视角，本公开内容的发明构思在于：各节点自建一种高分辨率的计时手段，计时本节点到达序，通过共享和分析各到达序的相对时间信息，得出还原序，再就还原序达成共识序，并将共识序的增长作为一种推进共识时间的手段，从而在不依赖于网络外部时间源的前提下实现一种网络内生时间并提供时间服务。

所述发明构思更为具体而言，第一，为节点提供一种计时手段使节点能记录下本节点的到达序和事务来源信息，从而避免信息损失；第二，为节点提供一种方法，使节点能根据各节点到达序信息和事务来源信息推理出事务进入共识网络的时序的信息，从而各自给出还原序；第三，为节点提供一种方法，使节点就各还原序达成共识序；第四，由共识序定义一种新的计时手段来辅助定序共识——从而形成一种新的事务定序共识方法、系统和设备；进一步地，为了避免节点在提供信息时的不诚实行为，提供一系列优选方法、系统和设备；进一步地，为了提高共识效率、降低共识成本，提供一系列优选方法、系统和设备。

由于本公开内容从信息和时间的视角定义了新的时间技术和事务定序共识技术，发明构思已不同于区块链技术，结合本公开内容中的技术特征，可以称本技术为“时间结绳”。当时间结绳技术与具体应用场景结合时，可以产生区块链技术所不具备的解决能力，例如，不依赖少数节点主观定序的去中心化证券交易所、不依赖人类时间的去中心化拍卖平台。

为了达到上述目的，本申请实施例采用以下技术方案：

第一方面，提供一种事务定序共识方法，所述方法包括：

获取待共识事务和相应的事务信息，所述事务信息包括事务编号、事务输入模块信息、事务入口节点信息，事务待执行信息和获取待共识事务的路径信息；

根据所述获取待共识事务的路径信息，将所述待共识事务划分为本地事务和外地事务；

验证、解析、同步所述待共识事务；

获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳；

同步不同共识节点之间的本地到达绳，得到同步后各节点到达序信息；

根据预设还原规则分析所述各节点到达序信息，得到本节点中两两事务的还原序以及一组事务连续的还原序信息，构建本地还原绳；

交换和同步不同共识节点之间的本地还原绳，得到同步后各节点还原序信息；

根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，所述共识绳包含已共识事务；

根据对所述共识绳解析，得到共识绳中已共识事务；

根据已共识事务更新状态数据。

在一种可能的实现方式中，更新状态数据之后，所述方法还包括：

共识节点根据共识绳的状态变化轨迹得到共识时间。

在一种可能的实现方式中，所述共识时间的当前时间包括当前时刻值和所述当前时刻值的校验值；

当前时刻值为所述共识序中当前已共识事务的个数；

获取所述共识时间的当前时间为时间戳。

在一种可能的实现方式中，在获取待共识事务和相应的事务信息之前，还包括：

获取事务触发信息；

获取共识网络中共识节点信息；

根据所述事务触发信息和共识节点信息，确定待共识事务的事务入口节点信息。

在一种可能的实现方式中，所述事务入口节点信息通过预设分配规则为所述待共识事务分配的入口节点，其中，所述预设分配规则包括指定、轮询、随机、就近和负载均衡；

根据所述事务入口节点信息区分所述待共识事务。

在一种可能的实现方式中，验证、解析所述待共识事务，包括：

根据预设验证项对所述待共识事务进行验证，得到验证结果；

当验证结果通过时，将所述待共识事务标记为已验证待共识事务；

根据预设解析规则对所述已验证待共识事务进行解析，得到解析结果；

将所述已验证待共识事务标记为已解析待共识事务。

在一种可能的实现方式中，同步所述待共识事务，包括：

获取所述已验证待共识事务及当前共识时间戳；

创建待共识事务同步消息，所述待共识事务同步消息包括待同步事务、消息发布时间戳、共识节点签名；

根据网络拓扑数据，按预设生成原则生成已验证待共识事务同步算法；

根据所述已验证待共识事务同步算法对所述已验证待共识事务进行同步，得到已同步待共识事务。

在一种可能的实现方式中，获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳；同步不同共识节点之间的本地到达绳，包括：

初始化共识节点的本地到达绳；

不间断地创建绳结且延长所述本地到达绳，所述绳结包括绳结编号、引入绳结数据和校验项；

当所述共识节点接收到本地事务时，将所述本地事务引入所述本地到达绳的当前绳结；

当在满足预设的本地到达绳的同步触发条件时，向本节点外共识节点同步所述本地到达绳中未同步部分；

根据预设到达规则接收、验证和同步外地到达绳；

解析所述本地到达绳和所述外地到达绳，得到同步后各节点到达序信息。

在一种可能的实现方式中，所述绳结编号为连续得自然数，且于所述绳结的产生时间相对应；

将所述本地到达绳作为本节点的计时手段；

所述本节点计时手段的当前时刻值为所述本地到达绳的当前的绳结编号。

在一种可能的实现方式中，验证外地到达绳的方法包括：

获取待验证外地到达绳；

根据预设到达绳验证项对所述待验证的外地到达绳段进行验证，得到验证结果；其中，所述预设到达绳验证项包括外地到达绳的签名、外地到达绳中每个绳结、每个绳结对应的事

务、每个绳结的校验值；

对所述待验证的外地到达绳段进行验证包括随机验证、并行验证和协作验证。

在一种可能的实现方式中，同步外地到达绳包括：

获取所述外地到达绳；

创建外地到达绳同步消息，所述外地到达绳同步消息包括待同步外地到达绳、消息发布时间戳、共识节点签名；

将所述外地到达绳写入所述外地到达绳同步消息，生成外地到达绳同步算法；

根据所述外地到达绳同步算法向其他共识节点发送外地到达绳同步消息，得到已同步外地到达绳。

在一种可能的实现方式中，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器和调度器，包括：

依次异步调用空闲的打结器进行打结操作；

生成绳结编号、确定前序绳结以及待映入绳结的事务，并创建引入绳结。

在一种可能的实现方式中，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器，且 N 个打结器各自有唯一的编号并持续竞争一个互斥锁，所述互斥锁携带一个整数变量 a 用于生成绳结编号和一个变量 b 用于记录最近一次竞争到互斥锁的打结器的编号；包括：

当一个打结器竞争到互斥锁后，将互斥锁所携带的整数变量 a 加 1 从而生成当前绳结编号；

读取互斥锁变量 b 从而获取上一个竞争到互斥锁的打结器的编号；

将变量 b 置为本打结器的编号，并判定是否有待引入绳结的事务；

当有待引入绳结的事务，则在所述本地到达绳上打事务结；

当没有待引入绳结的事务，则立即释放互斥锁并打空结；

其中，所述事务结包括本打结器的编号、上一个竞争到互斥锁的打结器的编号；所述空结为没有引入事务且已创建完毕的绳结。

在一种可能的实现方式中，所述本地到达绳的打结包括：

获取本地到达绳中含有当前结的绳段；

统计所述含有当前结的绳段中事务结的个数占所述含有当前结的绳段中绳结总数的占比；

当所述占比超过预设阈值时，增加本地到达绳的打结器；

当所述占比低于预设阈值时，减少本地到达绳的打结器。

在一种可能的实现方式中，同步不同共识节点之间的本地到达绳之前，包括：

对所述本地到达绳进行分段处理，得到到达绳段；

根据预设压缩规则对所述到达绳段的进行压缩处理；

所述压缩处理保留所述到达绳段上的事务结，并是删除所述到达绳段的非标记用空结。

在一种可能的实现方式中，两两事务的还原序以及一组事务连续的还原序信息，包括：

获取所述到达序信息中涉及的事务信息；

在所述到达序信息中涉及的事务中选择两个事务；

判断所述两个事务的还原序，输出所述两个事务的还原序信息和对所述两个事务的还原序信息的置信度；

迭代至获得所述到达序信息中一组事务连续的还原序。

在一种可能的实现方式中，交换不同共识节点之间的本地还原绳，包括交换还原序信息的共识节点组合：

获取共识节点信息，获取到达绳、还原绳和共识绳的作为初始值；

根据所述初始值生成随机数；

根据共识节点信息和所述随机数产生交换还原序信息的节点组合。

在一种可能的实现方式中，根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，包括：

获取本节点还原绳中未共识且已稳定的还原绳段；

接收其他 N 个共识节点的对应的请求未共识且已稳定的还原绳段；

根据已获取的到达序信息和还原序信息检查外地还原绳；

将接收的请求未共识且已稳定的还原绳段与本节点对应的共识且已稳定的还原绳段比较，保留不同绳段中相同率大于三分之二的时序关系；

重复与多个请求未共识已稳定的还原绳段比较，直到获得相同的时序关系，得到共识序信息，构建共识绳。

在一种可能的实现方式中，根据已共识事务更新状态数据，包括：

按照共识序信息解析共识绳中已共识事务；

根据所述已共识事务创建事务执行队列；

将所述已共识事务写入所述事务执行队列；

按队列处理规则，通过执行器对所述已共识事务执行队列中的事务进行处理。

在一种可能的实现方式中，执行器对所述事务执行队列中的事务进行处理，包括：

执行器获取所述已共识事务的待执行内容；

检查所述待执行内容；

将所述待执行内容转换为可执行指令；

当对应事务满足可执行指令的执行条件，按照可执行指令变更状态数据。

第二方面，提供一种事务定序共识系统，所述系统包括：

事务输入模块，用于产生待共识事务；

共识网络，用于对所述待共识事务分析处理，获得所述待共识事务的共识序；

状态获取模块，用于呈现状态数据的状态变化。

在一种可能的实施方式中，所述事务输入模块包括：

事务发起子模块，用于获取事务触发信息，

事务入口节点分配子模块，用于从共识网络中获取共识节点信息，根据预设入口节点分配规则、待共识事务和所述共识节点，确定所述待共识事务的入口节点；

事务发送子模块，用于将待共识事务传递给入口节点。

在一种可能的实施方式中，所述共识网络包括 N 个共识节点，所述共识节点包括：

事务处理子模块，用于处理本地事务和外地事务；

到达绳处理子模块，用于处理处理本地到达绳和外地到达绳数据；

还原绳处理子模块，用于处理本地还原绳和外地还原绳数据；

共识绳处理子模块，用于处理共识绳数据；

状态处理子模块，用于处理状态数据，并于所述状态获取模块进行数据交互；

时间处理子模块，用户获取时间戳，且验证所述时间戳。

在一种可能的实施方式中，所述事务处理子模块包括本地事务处理单元、外地事务处理单元、事务标记单元、事务存储单元；

所述本地事务处理单元包括本地事务获取子单元、本地事务验证子单元、本地事务解析子单元、本地事务同步子单元；

所述外地事务处理单元包括外地事务获取子单元、外地事务验证子单元、外地事务解析子单元、外地事务同步子单元；

所述事务标记单元，用于标记事务状态；

所述事务存储单元，用于存储本地事务、外地事务信息。

在一种可能的实施方式中，所述到达绳处理子模块包括本地到达绳处理单元、外地到达绳处理单元、到达序信息存储单元、到达序信息分析单元、到达绳处理优化单元；

所述本地到达绳处理单元包括本地到达绳初始化子单元、本地到达绳打结子单元、本地到达绳同步子单元、本地到达绳存储子单元；

所述外地到达绳处理单元包括外地到达绳获取子单元、外地到达绳拼接子单元、外地到达绳验证子单元、外地到达绳解析子单元、外地到达绳同步子单元、外地到达绳存储子单元；

所述到达序信息存储单元，用于存储本地和外地的到达序信息；

所述到达序信息分析单元，用于分析到达序信息；

所述到达绳处理优化单元，用于响应外部输入并优化到达绳的处理。

在一种可能的实施方式中，所述还原绳处理子模块包括到达序信息获取单元、本地还原绳生成单元、还原绳同步单元、本地还原序信息存储单元、外地还原绳同步单元，外地还原绳解析单元，外地还原绳验证单元，外地还原序信息存储单元、还原绳共识序生成单元。

在一种可能的实施方式中，所述共识绳处理子模块包括共识绳初始化单元、共识绳打结单元、共识绳事务验证单元、共识绳事务解析单元、共识绳事务执行单元、共识绳数据分析单元、共识绳同步单元、共识绳验证单元、共识绳状态输出单元。

在一种可能的实施方式中，所述状态处理子模块包括状态数据初始化单元、状态数据更新单元、状态数据验证单元、状态数据服务单元、状态数据存储单元、状态数据同步单元；

其中，所述状态数据服务单元包括状态数据订阅管理子单元、状态数据主动推送子单元、状态数据请求响应子单元、状态数据权限管理子单元。

在一种可能的实施方式中，所述时间处理子模块包括时间定义单元、时间解释单元、时间验证单元、时间标记单元、时间获取单元、时间同步单元。

在一种可能的实施方式中，所述状态获取模块包括状态呈现子模块、状态请求子模块、状态订阅子模块、状态合成子模块、状态存储子模块、状态分析子模块。

第三方面，提供一种共识节点设备，应用于第二方面所述一种事务定序共识系统，所述共识节点设备包括入口节点设备、到达序节点设备、还原序节点设备、共识序节点设备、状态节点设备、时间节点设备；

所述入口节点设备与所述到达序节点设备、所述状态节点设备数据连接；所述到达序节点设备与所述还原序节点设备数据连接；所述还原序节点设备与所述共识序节点设备数据连接；所述共识序节点设备与所述状态节点设备数据连接；

所述时间节点设备与所述入口节点设备、所述到达序节点设备、所述还原序节点设备、所述共识序节点设备、所述状态节点设备数据连接；

全节点设备具有入口节点、到达序节点、还原序节点、共识序节点、状态节点和时间节点的功能。

第四方面，提供一种共识应用程序，应用于第二方面所述一种事务定序共识系统，所述共识应用程序包括多个事务输入模块、多个状态获取模块和应用部署模块；所述应用部署模块，用于管理与共识应用相关的事务执行器和状态数据在共识节点的部署。

本申请实施例采用的上述至少一个技术方案能够达到以下有益效果：

1. 本申请所述共识节点构造本地到达绳记录事务到达本共识节点的时序的方法，使得共识网络保留了事务到达共识节点的时序证据信息，避免了信息损失，从而使得共识节点可以根据事务的到达序各自还原出事务进入网络的时序，共识节点的事务定序能力增强。

2. 本申请所述共识方法中不再设有主节点也不再需要竞选主节点，事务定序由共识节点共同完成，从而避免了因竞选主节点而造成的非理性竞争和资源浪费。

3. 本申请所述共识方法中，共识节点根据事务到达各节点的时序的证据，通过预设的客观分析方法完成事务定序，保证了定序过程的客观性，事务定序不再由某个节点主观操控，事务定序的依据不再是和事务时序无关的因素（如交易手续费高低），定序的结果也基本接近事务真实发生的顺序。同时，事务发起者获得了公平的记账待遇，不需要为了争取被记账而提高手续费，并且不再担心所发起的事务先进入网络反而长期不被记账或者后记账，从而丧失事务的时效性和先发优势的情况，于是增强了对共识网络的使用信心。

4. 本申请所述共识方法中，对于单股结绳来说，本地到达绳的一次打结时间约等于共识节点的一次哈希计算的时间，由于本地到达绳每次打结都可以看做时间推进，本地到达绳可以作为事务到达的计时器，这种设计就为事务到达计时提供了很高的时间分辨率；同时，共识绳也能为事务提供全排序。相比于比特币 10 分钟出一个块，而块中事务都视作同时发生，本申请不仅能实现事务的全排序，还能精确分析事务到达共识节点的频率变化，进而可以根据事务到达时序和共识时序进行大数据分析和优化处理。

5. 本申请所述共识方法中，事务被共识等同于事务被确认，事务共识不需要像比特币方案那样以出块间隔为周期批量完成，实现了共识的平滑推进，也不需要像比特币方案那样等待 6 个区块的时间才能确认一笔交易，极大缩短了事务的确认延迟，也就避免了因为提前人为确认一笔交易而被双花攻击的安全隐患。

6. 本申请所述共识方法中，共识节点必须不间断地构造到达绳，这种设计下，如果共识节点希望逆转记录两个事务的到达时序就必须花费相同的时间重新构造所涉及的绳段，并且同时要处理在重新构造绳段的过程中到达的事务，因此极大增加了共识节点恶意篡改事务时序的难度。尤其是采用多股结绳和变股结绳的时候更加难以伪造事务到达时序；恶意篡改共识绳时间和篡改事务到达时序的行为会在共识节点的到达绳处理阶段就被识别，从而无法影响到后续共识绳的生成，保证了共识系统的容错性。

7. 本申请所述共识方法中，共识节点通过客观分析事务到达共识节点的时序，还原出事务进入网络的时序，并对事务进入网络的时序达成共识，按照共识的时序执行事务，这种设计使得事务的执行时序逼近于事务进入共识网络的真实时序，在资源竞争性的场景中（如股票交易、能源交易、碳排放权交易、订票、知识产权抢注）实现了“先到先得”的交易公平性。

8. 本申请建立了一种去中心化的网络内生时间机制，其中时间推进不再由主节点完成，事务定序不再需要频繁变换参考系，避免了节点为垄断时间推进权而产生的中心化趋势。单一节点的到达序并非最终的共识序，从而避免了单一节点的到达序直接侵入共识序。

附图说明

为了更清楚地说明本申请实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

图 1 为本申请实施例一种事务定序共识系统的结构示意图；

图 2 为本申请实施例又一种事务定序共识系统的结构示意图；

图 3 为本申请实施例一种共识节点的结构示意图；

图 4 为本申请实施例另一种事务定序共识系统的结构示意图；

图 5 为本申请实施例一种事务输入模块的结构示意的图；

图 6 为本申请实施例一种状态获取模块的结构示意图图；

图 7 为本申请实施例包含共识节点设备的事务定序共识系统的结构示意图；

图 8 为本申请实施例事务输入模块的方法流程 x 图；

图 9 为本申请实施例一种事务输入模块发起待共识事务分配入口节点的方法示意图；

图 10 为本申请实施例一种事务定序共识方法的流程图；

图 11 为本申请实施例又一种事务定序共识方法的流程图；

图 12 为本申请实施例待共识事务数据结构示意图；

图 13 为本申请实施例验证、解析所述待共识事务方法的流程图；

图 14 为本申请实施例共识节点验证本地事务的方法示意图；

图 15 为本申请实施例共识节点验证外地事务的方法示意图；

图 16 为本申请实施例共识节点存储本地事务的方法示意图；

图 17 为本申请实施例共识节点获取外地事务的方法示意图；

图 18 为本申请实施例同步所述待共识事务的方法流程图；

图 19 为本申请实施例得到同步后各节点到达序信息的方法流程图；

图 20 为本申请实施例到达绳的结构示意图；

图 21 为本申请实施例绳结的结构示意图；

图 22 为本申请实施例绳结数据结构示意图；

图 23 为本申请实施例一种不间断地创建绳结且延长所述本地到达绳方法示意图；

图 24 为本申请实施例又一种不间断地创建绳结且延长所述本地到达绳方法示意图；

图 25 为本申请实施例共识节点接收外地到达绳的方法示意图；

图 26 为本申请实施例共识节点随机验证外地到达绳的方法示意图；

图 27 为本申请实施例共识节点并行验证外地到达绳的方法示意图；

图 28 为本申请实施例同步外地到达绳的方法流程图；

图 29 为本申请实施例共识节点存储外地到达绳的方法示意图；

图 30 为本申请实施例根据预设共识规则对本节点还原序信息进行共识处理得到共识序信息构建共识绳的方法流程图；

图 31 为本申请实施例根据已共识事务更新状态数据的方法流程图；

图 32 为本申请实施例执行器对所述事务执行队列中的事务进行处理的方法流程图；

图 33 为本申请实施例共识节点解析外地到达绳获知外地到达序的方法示意图；

图 34 为本申请实施例一种实体数据和关系数据的结构示意图；

图 35 为本申请实施例又一种实体数据和关系数据的结构示意图；

图 36 为本申请实施例共识节点根据从已获取的本地和外地到达绳中获取事务到达序信息的方法示意图；

图 37 为本申请实施例一种共识节点根据预设还原规则处理事务到达序信息输出两两事务的还原序和所有事务的还原序的方法示意图；

图 38 为本申请实施例共识节点构建本地还原绳存储还原序信息的方法示意图；

图 39 为本申请实施例共识节点多轮次交换还原序信息对本地和外地还原绳进行共识处理形成共识序的方法示意图；

图 40 为本申请实施例一种共识应用程序的结构示意图。

具体实施方式

为了使本技术领域的人员更好地理解本申请方案，下面将结合本申请实施例中的附图，对本申请实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本申请一部分的实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都应当属于本申请保护的范围。

需要说明的是，本申请的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的本申请的实施例能够以除了在这里图示或描述的那些以外的顺序实施。此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

下面对于背景进一步描述：

事务是变更状态的意志载体，状态是事务执行的现实结果。发起的是事务，获取的是状态。状态是发起新事务的依据，事务是产生新状态的动因。

只有被执行的事务才能改变状态，不同的事务执行顺序会产生不同的状态变化轨迹。一个状态被改变的机会是一种一次性资源。当一个状态诞生后，该状态被改变的机会窗口就打开，当该状态被改变到新状态后，原状态被改变的机会窗口就关闭，也就是当一个事务将原状态改变到新状态后，就不可能有另一个事务基于原状态进行另一次改变。于是，事务之间存在因果顺序，一个事务只能因应前一个事务造成的状态去创造新的状态。事务的因果顺序

决定事务的执行顺序，也就决定了状态变化轨迹。

意志未必能改变现实，确定了因果顺序的事务未必被执行。事务被执行需要满足两个条件，一是所基于的状态未被其他事务改变，二是事务的预期执行结果满足合理性。当两个事务相互冲突（希望基于相同的状态做出改变）时，就需要确定事务的因果顺序，前位事务被执行，后位事务被舍弃，例如同一块钱不能被花两次（特指“双花”）；当两个事务不预设所基于的状态时，后位事务因为要基于未知的新状态而可能丧失预期执行结果的合理性，从而不被执行，例如从只有一块钱的钱包不可能两次取出一块钱。任何主体都不希望自己的意志被舍弃或压制，因此，事务对因果顺序存在竞争需求，也就对事务定序机制提出要求。

理想的事务定序机制应当客观、公平、安全、稳定、高效。所谓客观，是指事务因果顺序应当基于客观证据而不被主观操纵；公平，是指不应当让事务在定序过程中获得特权；安全，是指已经被确定的事务因果顺序不应当被破坏；稳定，是指主体对事务的因果顺序有稳定的预期；高效，是指事务从发起到完成定序的延迟不应过长。

在共识网络中，数据在不同节点中存有副本，各节点接收并执行事务来变更数据副本的状态。为了使数据副本有一致的同步的状态变化轨迹，节点需要就事务因果顺序达成共识，并按照所共识的事务因果顺序来确定事务执行顺序。

主体的感知能力有限，不可能在第一时间知道世界上所有新发生的事。并且显然地，主体对事务定序的前提是获知待定序的事务，各主体对事务因果顺序达成共识的前提是至少形成了一个可理解和可验证的事务定序提案，而这个前提是存在一种公允的事务定序方法。

人类创造了一种公允的事务定序方法，将一个或一组可持续、可预测、不可逆、公共可见、变化足够频繁、不受人类控制的状态变化轨迹作为参考系，（即时间定义），设计一种计数系统对参考系的状态进行编号，（即时间解释），当参考系产生新状态时就要将当前编号更换为新状态对应的编号，（即时间推进），将事务分别与该参考系的状态编号建立映射，（即计时或盖时间戳），并以一种不易失和可举证的方式记录这些映射，（即时序举证），再以状态的先后顺序确定事务因果顺序，（即定序，时序确定）——这就是时间机制。

在物理空间中，人类将天文现象（如地球自转、地球公转、月球公转）作为参考系，通过政治博弈完成了人类时间的全球统一标准，由中心机构完成时间解释（即什么样的天体运动状态对应什么时刻）和时间发布（即授时），从而建立了一种中心化的人类时间机制。目前人类还不能明显操纵天体运动，因此人类暂时不掌握物理世界的时间推进权。

然而，网络空间中并没有物理空间的天文现象作为参考系，为摆脱对物理空间的依赖，就需要将且仅将网络内部活动（如计算、通信）所形成的状态变化轨迹作为参考系。网络空间是人造的，网络内部活动也是可以主观控制的，为了避免网络时间的解释、推进、发布权力被单一节点垄断或形成特殊优势，就需要建立不被少数节点操控的时间机制。

本申请所述的“事务定序共识”是指共识网络中各节点不信任外部时间，而仅通过网络内部的计算和通信手段形成不被少数操控的时间机制，对进入网络的一组事务的因果顺序达成不可逆的一致性共识，从而使各数据副本状态有一致的变化轨迹。

现有技术还存在后续问题：

出块机制将一组事务与同一个时刻绑定从而认为这组事务同时发生，降低了事务时序的分辨率（好比是认为所有赛跑运动员都同时撞线），也明显与事务的真实时序不符。

区块由单一节点向全网复制就存在区块同步延迟，系统通过设置出块难度来维护稳定的出块间隔时间为保证区块被同步到全网预留充足时间窗口，这也导致网络共识推进和事务的执行缺乏平滑性，这种出块间隔也给了攻击者充分的作恶时间窗口。

区块数据在多轮共识后才会在概率上被确认（例如比特币的交易确认一般要经过六个区块周期，平均每个周期 10 分钟），这导致事务确认延迟较高。这意味着网络状态数据长时间处于不确定性，这种不确定性给主体的决策造成风险。

现有的公平选主机制，只是“民主地选出一位能力低下的独裁者”。选主机制表面上实现了记账权分配的公平性，但对事务定序准确性、时效性和吞吐量的提高没有实质上的帮助，反而因为争夺记账权而造成算力和电力的浪费越来越严重。

上述这些后续问题会导致主体对于所提交的事务被如何定序及何时执行缺乏稳定的预期，对系统的公平性提出质疑，并承担越来越高的成本，从而不愿意使用这种系统。因此，现有技术局限于对事务定序实时性、公平性和吞吐量要求不高的业务场景。

造成现有技术后续问题的原因是，第一，节点无法第一时间对事务计时举证而导致各个节点的排序能力不足，第二，选主环节的设计使时间推进权、时间解释权和事务定序权同时集中在一个节点。因此解决问题的基本思路是，设计一套机制，在事务进入网络的第一时间就能形成可举证的计时信息，并且将时间推进权、时间解释权和事务定序权进行分离和分散，不再产生主节点。要想设计这种机制就需要对分布式系统的基础理论作出突破。

分布式系统的事务定序共识问题在上世纪 70 年代由 Lamport 开始讨论，但是受狭义相对论影响，研究者们把不同的节点理解为不同参考系中的观察者，并且认为不同参考系的观察者对事务的因果顺序有不同的认识，只能通过在不同参考系中传递事务因果顺序来形成共识，于是事务定序共识往往要先选定以哪一个参考系中的观察者所判断的事务因果顺序为准，也就是确定主参考系，对应选主；并将主参考系的观察者所判定的事务因果顺序强制推行到其他参考系，对应出块；作为整个网络对事务因果顺序的共识，对应上链。这种设计存在的问题包括：第一，给了单一参考系主观操控事务因果顺序的能力。事务定序共识方法的“公平性”被妥协为，不能由单一参考系一直垄断事务定序权，而是要设计一种机制来随机变换主参考系，但这并没有改变单一参考系在成为主参考系后对事务因果顺序的主观操纵能力，没

有根本上解决公平问题；第二，由某一参考系判定的事务因果顺序本身不具有排他性，只能由后续多轮次的参考系通过继承这种判定来逐渐巩固这种事务因果顺序的不可逆转性，于是事务因果顺序的最终确认必然存在明显的滞后性；第三，单一参考系对一组因果顺序的判定只能从该参考系出发向全网其他参考系传播，并且预留足够的传播时间使所述判定能尽量到达各参考系，必然导致一组事务的共识效率难以提升。

事实上，狭义相对论并不完全适用于网络空间。在不考虑量子技术的情况下，虽然绝对实时和绝对客观的排序共识也是不可能实现的，绝对时间也被认为不存在，但是在地球尺度上、在人类对事务定序准确度和时效性的可接受范围内、在现有计算和网络技术能力下，构造出一种接近绝对时间的事务定序共识系统并非是不可能的。

在这种思路下，选主就不再必要，确定一种客观公正的定序依据就成为新路径。在一个事务的生命周期中，事务由某个主体发起，从某个节点进入共识网络，再传播到达其他节点，然后形成事务因果顺序的共识。其中，事务发起的时机由主体自主控制，事务进入共识网络的时序是客观形成的，而事务经过物理网络到达其他节点的时序则不再可控，并且，事务进入网络的时刻表示事务的发起主体与共识网络的交互关系建立，即事务所承载的意志已经传达到了共识网络。因此，将事务进入共识网络的时序作为事务因果顺序是最为客观公正的。

在确定了事务定序的客观依据后，还要避免节点对事务定序的主观操控。至少有四层思路避免节点操纵事务因果顺序。第一，将节点操纵事务因果顺序的机会窗口压缩到小于节点操纵事务因果顺序所需的反应时间，也就是让节点不能操纵；第二，让节点操纵因果顺序所需的机会成本大于节点操纵事务因果顺序所能获得的预期收益，也就是让节点不敢操纵；第三，让节点不操纵事务因果顺序所能获得的收益大于操纵事务因果的预期收益，也就是让节点不想操纵；第四，不设主节点，让节点操纵的事务因果顺序只作为中间结果而无法侵入到最终结果，也就是让节点操纵无效，根本上杜绝操纵的负面影响。

事务定序共识的本质是消除事务因果的不确定性并形成对事务因果顺序的公共知识，也就是加工信息、输出知识，因而需要确定信息构成、信息来源、信息捕获方式、知识构成、信息向知识的转化方法。为提高知识输出的能力上限，就要提高信息输入能力，信息越充分，消除的不确定性就越多，就越能输出确定的知识。进一步的，同步、共识的目的就是要保持各节点有相同的信息和知识储备，这样各节点才能做出一致的行为。

在涉及选主的方案中，由于主节点的产生存在不确定性，并且主节点在输出知识时存在主观性，就这个整个网络来说，在知识输出环节引入了新的不确定性；主节点对于自己输出的知识有天然的认知优势，不仅引入了新的不公平，也可能引入新的不确定性；主节点输出的知识需要经过多个轮次才能“最终确认”，这个过程中存在较大被逆转的机会，也就长时间存在不确定性。这样就违背了通过定序共识来消除不确定性的初衷。

如果不设置主节点，就需要让每个节点都有足量的信息去生产出唯一相同的知识。通俗地讲，这个世界没有人能告诉你什么是绝对正确的，每个人都掌握着真相的碎片信息，每个人都可以共享碎片信息去帮助他人还原真相，每个人都会收集到足够的信息，并且按照相同的规则来处理信息，人们最终将得出相同的判断。

一组事务一般从多个节点进入共识网络，于是每个节点都无法直接测得所有事务进入共识网络的时序，但是每个节点都直接测得事务到达本节点的时序。这就需要找到一种方法，使各节点通过分析事务到达节点的时序信息判断出事务进入网络的时序信息，继而形成对事务进入网络的时序共识。第一，由于事务通过某一个共识节点进入网络，那么该节点就有能力判断该事务进入网络的时间必然晚于之前接收到的其他事务；第二，由于先进入网络的事务大概率先到达更多的共识节点，那么共识节点综合其他共识节点接收事务的顺序可以对比出哪个事务率先到达更多的节点，也就可以大概率正确判断事务进入网络的先后顺序；第三，节点间多轮次分享自己对事务进入网络的时序判断，并接受多数结果，最终可以收敛出对事务进入网络的时序的一致性判断，从而达成共识。

进一步地，已共识的事务序列可以作为时间参考系。序列中的事务按照客观规则进行编号，最新被共识的事务按照客观规则解释为当前时间，每新增一个共识事务视作一次时间推进，由于每个节点都按照相同规则维护相同的已共识的事务序列，时间定义权、时间解释权、时间推进权就分散于每个节点，无法被少数节点主观操控。这样就形成了客观的时间机制。利用客观的时间机制可用于事务进入网络时的计时，从而又增加了一个信息维度。

本申请所述事务定序共识方法的基本思路为：第一，把整个网络作为一个参考系，将事务进入网络的时序作为确定事务因果顺序的依据，各共识节点致力于就事务进入网络的时序达成共识；第二，共识节点各自维护一个时间参考系，根据这个时间参考系记录下事务到达本节点的时序证据，并及时向网络同步所述证据；第三，每个节点获取到事务到达各共识节点的时序证据，根据预设规则判断事务进入网络的时序；第四，各共识节点相互共享对于事务进入网络的时序判断，接受多数结果而形成对于事务进入网络的时序共识；第五，各节点将已共识的事务序列作为另一个时间参考系，从而对事务增加了一个计时维度。

本申请相关术语解释：

状态：在一个有记忆的系统中，一个或一组主体设定若干个参考项来描述一个事物，当这些参数项的值一定时，则称这个事物“处于一个状态”，当这些参考项中任意一个值发生了变化，或者参考项的个数发生了变化时，则称这个事物发生了“状态变化”。其中，

采用C表示可选的参考项的键名的集合，s 表示一个状态，由一组参考项的键值对表达，右上角（n）为状态编号。

$$C=\{c_0,c_1,\cdots,c_k,\cdots\}$$
$$s^{(n)}=\left(x_0^{(n)},x_1^{(n)},\cdots,x_k^{(n)}\right),\quad \forall x_i^{(n)}\in s^{(n)}\Rightarrow x_i^{(n)}=(c_j,v^{(n)}),c_j\in C$$

一个事物的状态从这个事物上一个状态变化而来，也就是每次状态变化都是从一个“始态”变为一个“终态”。采用S表示这种状态变化关系，而所述状态变化关系也是一种事物，因而S也可以表示所述状态变化关系作的状态，右上角的（n）为状态的编号。

$$S^{(n+1)}=\left(S^{(n)},s^{(n+1)}\right)$$

状态变化轨迹：当一个事物的状态不断变化，这个事物也随之产生新的状态变化关系，历次产生的状态变化关系按照产生的顺序形成一个序列，称为这个事物的“状态变化轨迹”，记作Ψ。一个事物的状态变化轨迹必然是单向延长的。

$$\Psi=S^{(0)},S^{(1)},\cdots,S^{(n)},\cdots$$

通过一组连续的自然数与状态变化轨迹中的元素建立一一映射的关系，自然数的大小顺序与状态变化轨迹中的元素产生的顺序一一对应。

进一步地，约定，状态展开集合是状态中所有参考项的集合，记作，

$$\widetilde{S}^{(n)}=\left\{x_0^{(n)},x_1^{(n)},\cdots,x_k^{(n)},\cdots\right\}$$

状态轨迹的展开集合是轨迹中所有元素的展开集合的元素的集合，记作，

$$\widetilde{\Psi}=\left\{x_0^{(0)},x_1^{(0)},\cdots,x_k^{(0)},\cdots x_0^{(n)},x_1^{(n)},\cdots,x_k^{(n)},\cdots\right\}$$

事件：将一个事物的一次状态变化称为一个事件，也把一组事件称为一个事件。

时间：时间是一个或一组主体对所在系统的状态变化的感受。当任意一个事物的状态发生变化，或者一个主体认为一个事物的状态发生了变化时，则被主体视为时间变化。

连续时间：因为现实中人类无法穷尽认识所有事物的所有变化，但是根据经验，事物一定在发生变化，所以人类认为时间是连续变化的，也就认为时间是有长度的。

时间流逝：由于一个事物的状态变化轨迹必然是单向延长的，由所述状态变化轨迹所定义的时间也必然是单向变化的，而当时间向前推进时，将一个事件标记为当前时间的机会窗口也不断关闭消失，因此用“流逝”来形容时间的变化。

时间参考系：由于现实中任何一个主体都无法穷尽认识所有事物的所有状态变化，为了简化了主体感知时间变化和描述时间感受的难度，一个或一组主体协定将且仅将一个或多个事物的状态变化轨迹作为时间参考系。时间参考系记作：

$$\Theta=\left\{\Psi_i\middle|i\in N,\Psi_i=S_i^{(0)},S_i^{(1)},\cdots,S_i^{(n)},\cdots\right\}$$

时刻：当时间参考系被当作一个整体时，时间参考系也有状态变化轨迹，则时间参考系的状态变化轨迹中的每一项是一个时刻。

时延：两个时刻定义的时间区间的时间长度。

时序：利用所述时间参考系的状态变化轨迹来标记事件发生时间或预期发生时间的顺序。已知事件 0 在时刻 i 和时刻 m 之间发生，且事件 1 在时刻 m 和时刻 n 之间发生，且已知时刻 j 早于时刻 m，则可以确定事件 0 早于事件 1 发生。

$$S^{(i)} \prec e_0 \prec S^{(j)} \wedge S^{(m)} \prec e_1 \prec S^{(n)} \wedge S^{(j)} \prec S^{(m)} \Rightarrow e_0 \prec e_1$$

时间戳：为了书写交流方便，采用符号与时间参考系的时刻一一对应，所述符号为时间戳。由于所述时间参考系状态变化轨迹不断延长，所述时间戳也需要有相应生成方式。根据时间参考系状态变化轨迹的时刻对应生成时间戳的方法，称为“历法”。

$$t_n \leftrightarrow n \leftrightarrow S^{(n)},\ n \in N$$
$$t_n = (n, \varphi(S^{(n)})),\quad s.t. \forall i \neq j \Leftrightarrow t_i \neq t_j$$
$$t_n \prec t_{n+1} \Leftrightarrow S^{(n)} \prec S^{(n+1)}$$
$$\Gamma = t_0, t_1, \cdots, t_n, \cdots$$

事件计时：将事件与一个时间戳建立关联的行为，则称为事件计时。

同时：如果一组事件与相同的时间戳建立关联，则认为这组事件同时发生。

时钟：一套模拟时间参考系状态变化且能直接或间接输出当前时间戳的设备。主体对于时间参考系的认识，也是一种对时间参考系的模拟，也可以视作一个时钟。

时间分辨率：一个时钟所能表达的最小时延，称为所述时钟的时间分辨率。时间分辨率体现了时钟帮助主体分辨时间变化和事件时序的能力。

时间定义：当时间参考系和历法确定以后，主体对于时间的理解就确定了，因此把确定时间参考系和历法的行为称为“时间定义”。

时间解释：当时间被定义之后，把确定当前时间的行为称为“时间解释”。

时间推进：在确定当前时间后，把产生当前时间戳的行为，称作“时间推进”。

时间发布：也称“授时”。

时间接受：也称“受时”。

人类时间：人类定义的时间。例如，人类将地球公转、月球公转、地球自转的状态变化轨迹集合作为时间参考系，对应记作 X 年、阴历 X 月、X 日。(为了使年月日的周期关系稳定，引入了闰年闰月机制)。再利用机械、晶振模拟时间参考系的变化来制作时钟。

网络时间：将网络内部的事物状态变化轨迹作为时间参考系所定义的时间。进一步地，网络内生时间，指的是，相对于人类依靠天文观测而建立的历法时间（例如根据地球公转、月球公转、地球自转来建立的年、月、日历法）来说，网络内生时间是依赖且仅依赖于在网

络内部产生的计算和通信事件而标记和推进的时间。采用网络内生时间可以摆脱对网络外部时间源的依赖。而去中心化的网络内生时间指的是时间不由某个中心控制和授时，而是由网络内的节点通过某种分散的机制共同推进的时间。

事务：主体所产生的变更某个目标事物状态的意志的载体。主体通过发起事务来表达自己变更某个事务状态的意志，当事务被接受且满足执行条件后就会被执行，事务的执行会改变目标事物的状态。

$$T: s^{(n)} \rightarrow s^{(n+1)}$$

$$T: S^{(n)} \rightarrow S^{(n+1)}$$

强/弱事务：一个事务能否被执行存在不确定性，主体在发起事务时往往并不能确定事务执行所对应的始态和终态。把明确表达了始态和终态的事务称为强事务，否则为弱事务。

事务输入模块：向系统输入事务的模块。在具体表现形式上，可以是手机客户端、网页、设备板卡、服务器应用、无人机、智能汽车等任何具备向共识网络输入事务的功能的整体或部分，可以从设备角度理解，也可以从程序角度理解，具体形态和位置不做限定。本申请中的“事务输入模块”可能指一个模块，可能指一组模块，可能指所有的事务输入模块，具体的含义会配合语境确定。

状态数据：状态数据是事务执行的结果，状态数据可供状态数据获取模块获取，在本申请中，状态可以指状态数据集合，可以指状态数据集合的子集，可以指状态数据集合中的特定元素。例如，在“为用户 A 初始化账户 1”事务执行后，在状态数据集合中会新增一个编号为 1 且余额为 0 的账户实体数据，并新增一个表示“用户 A 拥有账户 1”的关系数据。之后，在“用户 B 向账户 1 转入 1 元钱”事务执行后，账户 1 的余额会从 0 增加到 1 元。进一步地，共识应用和共识节点的程序代码也是一种状态数据，程序代码的内容、版本、开发者、所有者、执行者信息也可以在状态数据集合中存储，也可以通过事务执行来更新状态。

状态获取模块：从系统获取状态的模块。可以是手机客户端、网页、设备板卡、服务器应用、无人机、智能汽车等任何具备从共识网络获取状态的功能的整体或部分，可以从程序角度理解，也可以从设备角度理解，可以和事务输入模块组成一个程序或一个设备，也可以与事务输入模块分布在不同的程序或设备，具体形态和位置不做限定。本申请中的“状态获取模块”可以指一个模块，也可以是一组模块，也可以指所有状态获取模块，结合语境可以确定。状态获取模块从共识网络中获取状态，至少包括两种方式，包括：

方式一状态获取模块向共识网络发送获取请求，共识网络响应请求并将状态返回给状态获取模块。具体的，状态获取模块可以每次向随机选定的一个或多个共识节点发送获取请求；也可以根据预设的选定规则每次向一个或多个共识节点发送获取请求。

方式二状态获取模块预先向共识网络订阅状态，共识网络根据订阅需求主动向状态获取模块推送状态；具体的，状态获取模块可以向一个或多个共识节点订阅相同主题的状态；也可以向一个或多个共识节点订阅不同主题的状态数据；

例如用户A和用户B分别向共识网络发送对电影院座位C的订票选座请求:T(A)和T(B)；共识网络获取到 T(A)和 T(B)，经过共识处理后得出时序：T(A)早于 T(B)；按照顺序先执行T(A)，由于此时C未被预定，将座位C的状态改为“已被A预定”，再执行T(B)，由于座位C已被预定，T(B)不改变座位C的状态；用户A和用户B分别到获取C座位的状态，其中，用户A每过一段时间就向共识网络请求一次座位C的状态，而用户B订阅了座位C的状态，当座位C的状态被改变时，共识网络主动向用户B推送座位C的最新状态；

主体：拥有状态获取能力和事务输出能力的个体或一组个体的整体。一个主体一般包括若干个事务输入模块和若干个状态获取模块，并有一个数据库存储已获取的系统数据，一个缓冲区用于存放对系统数据的处理结果，并有一组决策模块用于分析理解系统状态并决定发起事务。M表示主体，I表示事务输入模块集合，F表示状态获取模块集合，Σ表示以获取的系统状态的集合，B表示缓冲区，D表示决策模块集合。

$$
\begin{aligned}
&M=(I,F,\Sigma,B,D),\\
&I=\{i_0,i_1,\cdots,i_m,\cdots\},\\
&F=\{f_0,f_1,\cdots,f_m,\cdots\},\\
&\Sigma=\{x_0,x_1,\cdots,x_m,\cdots\}\subseteq\widetilde{\Psi},\\
&D=\{d_0,d_1,\cdots,d_m,\cdots\},\\
&B=\left\{d(\Sigma'\cup B')\middle|d\in D,\Sigma'\subseteq\Sigma,b\subseteq B'\right\}
\end{aligned}
$$

一个主体已获取的系统数据是系统状态变化轨迹的展开集合的子集，未必包含系统的当前状态，也不一定完整。也就是主体对系统的认识往往是片面、滞后、残缺的。

事务因果顺序：当事务1基于的始态是事务0直接或间接造成的终态，则两个事务构成因果关系。一组事务按照因果关系排列成事务因果顺序。

$$O_{ce}=T_0\prec T_1\prec T_2\prec T_3\prec\cdots$$

事务执行顺序：事务被实际执行或计划执行的顺序，一般由事务因果顺序决定。

$$O_{ex}=T_0\prec T_1\prec T_2\prec T_3\prec\cdots$$

事务定序：一种流程，输入一组事务并输出所述事务之间的因果顺序。

共识节点：简称节点，一种特殊的主体，相互直接或间接连接成一个网络，对网络内一组事务进行定序，并就这组事务的因果顺序和执行顺序达成共识。共识节点可以主动发起事务，也可以接受其他主体的委托向网络内输入事务，也可以同步其他节点发来的事务。

所述共识节点可以理解为程序、设备或系统，从设备角度而言，可以包括但不限于以下设备：个人计算机、大中型计算机、计算机集群、手机、平板电脑、智能可穿戴设备、车机、智能包装、飞行器、航行器等。

共识网络：简称网络。由一组数量可变的共识节点组网而成，共识节点可以按照预设的规则加入或者退出共识网络。所述事务输入模块向共识网络输入待共识事务，具体来说就是事务输入模块向共识网络中的共识节点输入待共识事务。组成一个共识网络的共识节点之间不一定两两直接相连，但是每两个共识节点之间都有至少一条通路。

用 G 表示共识网络拓扑，V 表示共识网络的节点集合，E 表示节点连接关系集合。例如，一个拥有 4 个共识节点的共识网络表示为，

$$G=(V,E)$$
$$V=\{n_0,n_1,n_2,n_3\}$$
$$E=\{\langle n_0,n_1\rangle,\langle n_0,n_2\rangle,\langle n_1,n_2\rangle,\langle n_1,n_3\rangle,\langle n_2,n_3\rangle\}$$

入口节点：主动发起事务或接受其他主体的委托向网络内输入事务的共识节点，称为所述事务的入口节点。一个事务输入模块向共识网络的某个入口节点输入事务，被表达为，

$$i:T^{(n)}\mapsto n, i\in I\in M, n\in V$$

事务进入网络：一个事务被入口节点完成计时并签名之后从入口节点向其他节点发出，视作该事务进入了共识网络。本申请提供的实施例假设事务被入口节点盖时间戳后立即签名并立即发出，于是用事务的共识时间戳表示事务进入网络的时间。

事务同步：一个过程，即事务在节点之间转发，以尽量使得网络内所有节点都获取到所述事务。事务在节点之间的转发关系表达为，

$$n_j:T^{(n_i)}\mapsto n_k,\ n_i,n_j,n_k\in V, j\neq k, i\neq k$$

事务到达节点：一个事务以本节点为入口节点或者被其他共识节点转发向本节点，被节点获取，视作事务到达了本节点。本申请提供的实施例假设事务被节点获取后被立即处理并填写本地时间戳和共识时间戳并立即引入本地到达绳，于是用事务的本地时间戳和共识时间戳表示事务到达本节点的时间。

共识：共识网络中的节点就一个提案达成相同的认识。

同步：指的是共识节点将待同步内容写入同步消息并将所述同步消息按照一定的方式（同步算法）传播到其他共识节点，其目的是让某些最初被单一或少数共识节点掌握的信息最终扩散为被指定范围内的所有共识节点（一般是共识网络内的全部共识节点，也有可能是符合预设条件的一部分共识节点）都知晓。

所述同步消息，至少包括：

消息发布时间戳，所述时间戳为消息发布时共识绳中的末尾绳结的序号和哈希值；

待同步内容，可以是待同步的本地事务、待同步的本地到达绳（可以是已压缩的）；

消息的发布者共识节点的签名。

所述同步算法可以是 PBFT、雪崩、Gossip 等，也可以是共识节点自主约定的算法，本申请不做具体限定。

事务定序共识：共识网络中的节点就一组事务的因果顺序达成一致的认识。事务的因果顺序决定事务的执行顺序，而事务执行顺序的不同会产生不同的状态变化轨迹，为了保证状态有一致的变化轨迹，共识网络需要对事务的执行顺序达成共识。

例如，有 3 个事务（T1：A 向 B 转 5 元；T2：A 向 B 转 3 元；T3：A 向 B 转 10 元。其中 A 的钱包初始余额为 8 元，B 的钱包初始余额为 0 元），规定当事务中待转出金额超过余额时则执行失败，则，如果执行顺序为 T1、T2、T3，则（A 余额,B 余额）的状态变化轨迹为（8,0）（3,5）（0,8）（0,8），而如果执行顺序为 T3、T1、T2，则状态变化轨迹为（8,0）（8,0）（3,5）（0,8）。本申请中的“事务”可能指一个事务，可能指一组事务，可能是事务数据，也可能指事务请求，也可能是事务执行过程，具体的含义会配合语境确定。

从程序角度而言，事务定序共识流程的执行主体可以是应用(APP)、Web 浏览器、个人计算机(PC)端程序、嵌入式设备程序等。从设备角度而言，该流程的执行主体可以包括但不限于以下设备：个人计算机、大中型计算机、计算机集群、手机、平板电脑、智能可穿戴设备、车机、智能包装、飞行器、航行器等。

事务共识延迟：一个事务从进入网络的时刻到被写入共识绳的时刻所经历的时延。

待共识事务：事务编号、事务输入模块标识、入口节点标识、事务输入模块对系统指定状态数据的变更请求的描述。所述事务输入模块可能是用户客户端，也可能是其他设备，这里不做具体限定。系统通过密码学手段实现对事务内容是否被篡改的可验证性。其中，事务待执行内容中一般包括：用户 ID、指定的事务执行器 ID、待执行状态更新描述指令。

本真序：事务输入模块各自向共识网络输入事务，客观上产生了事务进入共识网络的时序，称为“本真序”。共识网络是分布式的，事务输入模块输入的事务会以共识网络中不同的共识节点为入口进入共识网络，在现有技术条件下不存在某个时钟能立即记录下所有事务进入网络的时序，而共识节点也不会以任何外部的时间戳作为事务排序的依据。因此，所述的本真序不会在事务进入网络后立即被共识网络的所有共识节点获取。记作，

$$O_T = T_0 \prec T_1 \prec T_2 \prec T_3 \prec \cdots$$

到达序：共识网络中的各共识节点获取事务并向其他共识节点同步事务，客观上产生了事务到达各共识节点的时序，称为“到达序”。各共识节点通过各自构建的时钟记录下到达序。所述事务“到达”一个共识节点的方式包括：事务以所述共识节点为入口共识节点从而到达了所述共识节点，或者事务进入共识网络后经过同步机制被传播到了所述共识节点。显然，各共识节点记录的到达序可能不同，到达序也不等同于本真序。打比方说，池塘上的浮标有两种方式知道一个雨滴落在了池塘上，一种是雨滴刚好落在了浮标上，另一种是雨滴落在水面上后散出来的水波推进到了浮标。所述到达一般指的是首次到达。记作，

$$\mathrm{O}_A^{n_0} = T_0^{(n_0)} \prec T_1^{(n_1)} \prec T_2^{(n_4)} \prec T_3^{(n_0)} \prec \cdots, \quad n_i \in V$$

还原序：共识节点之间同步各自的到达序记录，通过预设规则分析出每个到达序记录中蕴含的部分本真序信息，并将各部分本真序信息整合，从而各自还原出对本真序的判断。所述共识节点还原出的对本真序的判断，称为“还原序”。记作，

$$\mathrm{O}_R^{(n_0)} = T_0^{(n_0)} \prec T_1^{(n_1)} \prec T_2^{(n_4)} \prec T_3^{(n_0)} \prec \cdots, \quad n_i \in V$$

共识序。共识网络对一组事务的还原序的一致认识。共识节点之间共享各自的还原序，通过预设的共识处理方法形成对本真序的一致性判断。一般的，共识序一旦形成后不允许逆转，只能在末端加入新的共识事务而单向延长。于是，共识节点可以将共识序的一次单向延长作为共识网络的一次时间推进，即把共识序作为共识网络公用的时钟，共识节点可以利用共识序记录时间。共识序记作，

$$\mathrm{O}_C = T_0 \prec T_1 \prec T_2 \prec T_3 \prec \cdots$$

共识应用：与共识网络和用户进行交互，满足特定应用场景需求的主体。例如：在一个共识系统中包括了共识网络和若干共识应用，其中共识应用包括：电影订票应用、股票交易应用、物品拍卖应用、检测试验应用、无人机群应用，在每个应用中都包括事务输入模块、状态获取模块。例如：电影订票应用会输入“订票”“退票”“取票”等事务，也会获取“影院”“场次”“座位”等状态数据。从设备角度而言，每个应用都可以是整体或部分的形式安装在一个或多个设备上。例如：用户 A、B、C、D 的手机上都安装有电影订票应用 APP，也可以通过网页打开应用，在电影院大厅里安装有电影订票取票触控屏。

结绳：一种数据结构，包括：一组连续生成的数据区块，其中，每个数据区块在上一个区块生成后立即开始创建，且依赖所述上一个区块的数据（一般是将所述上一个数据区块的哈希值写入本区块），且每个数据区块有连续的编号。记作：Φ

绳结：结绳中的每个数据区块称作一个绳结。所述绳结的结构包括：

绳结编号，一般为连续的自然数，且在同一个到达绳中后产生的绳结编号要比之前生成的绳结的编号大。例如：第一个绳结的编号为 0，第二个为 1，第 N 个为 N-1。

前序结引用，一般为同一个本地到达绳中紧邻本绳结的上一个绳结的数字摘要值。例如，当前的绳结编号为 9，则前序结为 8 号绳结，前序结引用为 8 号绳结的哈希值；

事务引用，一般为一个待共识且已验证的事务的数字摘要值，表示引入了所述事务，或者为空值或特定值，表示本绳结没有引入事务；

校验项，可选的，为了增加到达绳的重建难度并易于检查被篡改的绳结，可以设置零个或多个校验项，所述校验项可以是当前绳结所见证的一些特征状态值，例如：

（1）到当前绳结为止所在的本地到达绳中引入的事务的总数；

（2）当前绳结的上一个引入了事务的绳结的编号；

（3）当前绳结向前一定数量的绳结中引入的事务的总数；

（4）当前绳结引入的事务的入口共识节点的 ID；

（5）当前绳结之前第 N 个满足某特征的绳结的数字摘要值；

（6）其他预设的特征值。

校验规则编号，可选的，为了灵活地在不同绳结中增减或变更校验项，可以设置校验规则的编号，指定每个校验项按照哪个校验规则来校验。从程序的角度来看，所述校验规则为一段可以验证所述校验码真伪的程序，在共识节点中预设了一组校验规则，给每个规则设置一个编号，那么通过绳结中的校验规则编号即可知道执行那段校验规则。

在本申请中，一个绳结可以用一个结构体表示，存储在内存中，也可以用 JSON、XML 等形式存储在文件、数据库、磁盘中，只要有预设格式保存上述编号、前序结引用、事务引用、校验项等内容即可，对绳结的格式和存储位置不做限定。

绳结的符号记作：K

打结延长：一个流程，为一个结绳新创建一个绳结。

引入绳结：将某个内容 X 作为打结的输入称作“将 X 引入结中”。

绳段：一个结绳中连续的一组绳结组成的集合称作一个绳段。例如，在一个本地到达绳中，存在 0-100 号绳结，其中，0-10 号绳结是一个绳段，89-93 号绳结也是一个绳段。

初始结：一个时间结绳初始化生成的第一个绳结（一般编号记作 0）。一个本地到达绳初始化时创建的位于最开始的绳结，就是所述本地到达绳的初始绳结，初始绳结在所述本地到达绳中没有前序结。

当前结：在一个持续延长的结绳中，当前正在创建的绳结，当前结的编号已经生成。或者是当前正在考察的绳结。

开头结：一个绳段中最早创建完毕的绳结。

末尾结：一个绳段中最晚创建完毕的绳结。例如，在 99-106 号绳结组成的绳段中，第 106 号绳结为最后一个生成的绳结，也就是末尾绳结。

事务结：引入了事务且已创建完毕的绳结。例如，事务结的“事务引用”属性值为被引入事务的数字摘要。例如，编号为 i，引入事务 T 的事务结记作：

$$K_{(T)}^{(i)}$$

空结：没有引入事务且已创建完毕的绳结。例如，空结的“事务引用”属性值为空或者表示未引入事务的特定值。例如，编号为 i 的空结记作：

$$K_{(\varnothing)}^{(i)}$$

前序结：如果绳结 0 的哈希值被引入绳结 1，则绳结 0 为绳结 1 的前序结。

后继结：如果绳结 0 的哈希值被引入绳结 1，则绳结 1 为绳结 0 的后继结。

时间结绳：一种用于定义时间和事件计时的结绳，所述结绳的状态变化轨迹作为时间参考系。在本申请实施例中存在两种时间结绳，到达绳和共识绳。时间结绳蕴含的信息本质是时间戳集合、事务集合、时间戳的顺序关系集合、事务与时间戳的绑定关系集合组成的四元组，其中，一个时间戳可以绑定 0 个或 1 个或多个事务。区块链是时间结绳的一种特例，其中，一个区块实际上就是一个时间戳和绑定这个时间戳的事务的集合。

$$\Phi \to \left(\Lambda, T, \left\{\left\langle t_i, T_j \right\rangle \middle| t_i \in \Lambda, T_j \in T\right\}, \left\{t_i \prec t_j \middle| t_i \in \Lambda, t_j \in \Lambda\right\}\right)$$
$$\Lambda = \left\{t_0, \cdots, t_n\right\}$$
$$T = \left\{T_0, \cdots, T_m\right\}$$

当时间结绳中的一个时间戳至多绑定 1 个事务，并且任意两个时间戳之间的时序都是确定的，并且不存在多个时间戳的时序形成环状结构，那么这个时间结绳就支持事务的全排序，也就是可以确定任意两个事务之间的时序关系。

到达绳：是用于承载到达信息的数据及数据结构的一种可能的实现方式，一种由共识节点构造，定义本地时间，并记录事务到达序的时间结绳，是一种共识节点以结绳方式记录的事务到达时序数据。由于所述结绳记录了事务到达共识节点的时序证据，故称作所述结绳为“到达绳”。其中，一般的，一个共识节点只构造一条到达绳。例如，一条由节点 n0 构造的本地到达绳可以被记作：

$$\Phi_A^{(n_0)} = K_{(T_0)}^{(0)} \prec K_{(T_1)}^{(12)} \prec K_{(T_2)}^{(60)} \prec K_{(T_3)}^{(130)} \prec K_{(T_4)}^{(400)} \prec K_{(T_5)}^{(425)}$$

其中，空结被省略不计，只列出事务结。

所述到达绳相比于比特币方案的交易缓冲区，至少增加了事务到达共识节点、事务到达

共识节点的时间、事务输入模块三个信息维度，可以极大地扩展了信息表达能力。

到达绳不仅能反映事务到达共识节点先后时序，还能一定程度上反映事务到达共识节点的频率变化，进一步还能反映当前网络拥堵情况，还能近似反映不同共识节点的相对打结频率，还能反映出本地事务晚于此前到达的外地事务进入网络。

从安全性角度讲，由于结绳的每个绳结都是前向依赖的，如果想逆转两个事务的顺序就必须重新花费时间构造所涉及的绳段，而在构造的过程中又需要把不断新接收的事务引入结绳，否则会积压多个待引入的事务而不得不连续地引入事务，这种伪造行为会在结绳上留下痕迹并且很容易被其他节点发现。然而，单一节点对事务到达时序的伪造难以影响到整个网络最终对事务进入网络时序的共识。可见，节点伪造到达绳是徒劳的。

本地到达绳：由本节点构造的到达绳，记录了事务到达本共识节点的时序数据。

到达绳同步：一种流程，使到达绳数据尽可能到达所有的节点。

外地到达绳：由其他节点构造的且被同步到本节点的到达绳。外地到达绳是本共识节点获取的由其他共识节点发来的事务到达所述发出到达绳数据的共识节点的时序数据。

本地到达绳未同步绳段：在一个本地到达绳中未被写入同步消息进行同步的绳段。

打结器：用于创建绳结的装置、程序、设备。

到达绳的并行度。

多股到达绳：一种在构造过程中并行度固定不变且并行度大于 1 的到达绳。

变股到达绳：一种在构造过程中并行度可以自适应调整的到达绳。

还原绳：一种由共识节点构造，记录事务还原序的数据结构。例如，由节点 n0 构造的地到达绳可以被记作：

$$\Phi_R^{(n_0)}$$

本地还原绳：由本节点构造的还原绳。

外地还原绳：由其他节点构造的还原绳。

本地还原绳的稳定绳段：在本地还原绳中，一组被判断连续的事务构成的序列，且经判断不可能有额外的事务被判断加入到所述序列的非末尾位置，则视作所述序列已稳定，用以表达所述序列的还原绳被称为本地还原绳的稳定绳段。

还原绳中的冗余时序数据。

共识绳：一种由共识节点构造，定义共识时间，记录事务共识序的时间结绳。

本地共识绳：由本节点构造的共识绳。

外地共识绳：由其他节点构造的共识绳。

共识绳同步：一种流程，使本地共识绳数据更新到最新状态。

本地共识绳未执行绳段：在本地共识绳中，从最早写入所述本地共识绳的已共识但未被解析执行的绳结到所述本地共识绳的末尾绳结所组成的绳段。

事务执行器：用于执行事务中待执行内容的装置、程序、设备。

事务待执行队列：一个队列，队列元素为待执行的事务。

本地时间：由本地到达绳的状态变化轨迹作为时间参考系定义的时间。

共识时间：由共识绳的状态变化轨迹作为时间参考系定义的时间。在共识网络中，已共识事务的数量是单向递增的，且每共识一个事务都意味着一段时间的流逝，则可以在共识系统中设置一种以已共识事务数量为基础的时间，称为共识时间，一种简单的共识时间戳表示方法是直接用一个整数表示，所述整数为当前已共识事务的数量。

状态合成：将从一个或多个共识节点获取的状态数据做合成处理，形成节点对于状态数据的自主认知。由于本申请所述共识系统中，各共识节点的共识进度会有细微差别，状态数据获取模块从单一共识节点所获取的状态数据可能不是最新数据、可能由于节点故障无法获取数据，为保证获取到最新数据以及保证获取能力，状态数据获取模块可能同时从多个共识节点中获取状态数据，再由状态合成模块从获取的多个状态数据中选择可接受的状态数据，发送给状态数据呈现模块进行处理。

时序还原度：一种度量指标，即通过事务定序共识流程所输出的共识序与所述事务的本真序之间的相似度，反映的是事务定序共识系统多大程度上还原了本真序。

去中心化交易所：一种程序、设备或系统，搭建运行在一种去中心化网络上，其中，交易时序不被网络节点主观操控，已形成的交易记录不可被篡改内容和逆转时序。在交易所场景中，交易时序严重影响交易结果。一般的，任何一笔交易都不希望自己客观产生的时序被任何人操纵，而交易通过设置更高的交易手续费来获取优先交易权也会被认为是不公平行为，高额手续费也会侵占交易者的利益并产生恶性循环的“剧场效应”。因而，现有的选举主节点进行主观定序的区块链方案不能满足对交易时序公平性有严苛要求的去中心化交易的场景。

本申请目的在于，发明一种事务定序共识方法、系统和设备，使得共识网络中各节点不信任外部时间，而仅通过网络内部的计算和通信手段形成不被少数操控的时间机制，对进入网络的一组事务的因果顺序达成不可逆的一致性共识，从而使各数据副本状态有一致的变化轨迹，其中，对于所述达成共识的事务因果顺序应尽量与所述事务真实进入网络的时序相同。

下面结合附图对本申请做进一步详细描述：

实施例 1

本申请实施例提供一种事务定序共识系统，如图 1 和图 2 所示，所述系统包括若干个事务输入模块、一个共识网络、若干个状态获取模块。

事务输入模块，用于产生待共识事务；共识网络，用于对所述待共识事务分析处理，获得所述待共识事务的共识序；状态获取模块，用于呈现状态数据的状态变化。

其中，如图 3 所示，所述共识网络包括 N 个共识节点，所述共识节点包括：事务处理子模块、到达绳处理子模块、还原绳处理子模块、共识绳处理子模块、状态处理子模块、时间处理子模块；其中：

事务处理子模块，用于处理本地事务和外地事务；所述事务处理子模块包括本地事务处理单元、外地事务处理单元、事务标记单元、事务存储单元；其中：

所述本地事务处理单元包括本地事务获取子单元、本地事务验证子单元、本地事务解析子单元、本地事务同步子单元；其中：本地事务获取子单元，用于获取本地事务；本地事务验证子单元，用于验证本地事务；本地事务解析子单元，用于解析本地事务；本地事务同步子单元；用于同步本地事务。

所述外地事务处理单元包括外地事务获取子单元、外地事务验证子单元、外地事务解析子单元、外地事务同步子单元；其中：外地事务获取子单元，用于获取外地事务；外地事务验证子单元，用于验证外地事务；外地事务解析子单元，用于解析外地事务；外地事务同步子单元；用于同步外地事务。

所述事务标记单元，用于标记事务状态。

所述事务存储单元，用于存储本地事务、外地事务信息。

到达绳处理子模块，用于处理处理本地到达绳和外地到达绳数据；所述到达绳处理子模块包括本地到达绳处理单元、外地到达绳处理单元、到达序信息存储单元、到达序信息分析单元、到达绳处理优化单元；其中：

所述本地到达绳处理单元包括本地到达绳初始化子单元、本地到达绳打结子单元、本地到达绳同步子单元、本地到达绳存储子单元；其中，本地到达绳初始化子单元，用于初始化本地到达绳；本地到达绳打结子单元，用于本地到达绳的打结延长；本地到达绳同步子单元，用于同步本地到达绳；本地到达绳存储子单元；用于存储本地到达绳。

所述外地到达绳处理单元包括外地到达绳获取子单元、外地到达绳拼接子单元、外地到达绳验证子单元、外地到达绳解析子单元、外地到达绳同步子单元、外地到达绳存储子单元；

其中，外地到达绳获取子单元，用于获取外地到达绳段；外地到达绳拼接子单元，用于拼接获取的外地到达绳段；外地到达绳验证子单元，用于验证外地到达绳；外地到达绳解析子单元，用于解析外地到达绳获取外地到达绳的到达序信息；外地到达绳同步子单元，用于同步外地到达绳；外地到达绳存储子单元，用于存储外地到达绳。

到达序信息存储单元，用于存储本地和外地的到达序信息。

到达序信息分析单元，用于分析到达序信息。

到达绳处理优化单元，用于响应外部输入并优化到达绳的处理。

还原绳处理子模块，用于处理本地还原绳和外地还原绳数据；所述还原绳处理子模块包括到达序信息获取单元、本地还原绳生成单元、还原绳同步单元、本地还原序信息存储单元、外地还原绳同步单元，外地还原绳解析单元，外地还原绳验证单元，外地还原序信息存储单元、还原绳共识序生成单元。

共识绳处理子模块，用于处理共识绳数据；所述共识绳处理子模块包括共识绳初始化单元、共识绳打结单元、共识绳事务验证单元、共识绳事务解析单元、共识绳事务执行单元、共识绳数据分析单元、共识绳同步单元、共识绳验证单元、共识绳状态输出单元。

状态处理子模块，用于处理状态数据，并于所述状态获取模块进行数据交互；所述状态处理子模块包括状态数据初始化单元、状态数据更新单元、状态数据验证单元、状态数据服务单元、状态数据存储单元、状态数据同步单元；其中：

状态数据初始化单元，用于初始化状态数据处理模块和状态数据。

状态数据更新单元，用于更新状态数据。

状态数据验证单元，用于验证状态数据。

状态数据服务单元包括状态数据订阅管理子单元、状态数据主动推送子单元、状态数据请求响应子单元、状态数据权限管理子单元。其中，状态数据订阅管理子单元，用于管理状态获取模块对状态数据的订阅；状态数据主动推送子单元，用于主动向状态获取模块推送状态数据；状态数据请求响应子单元，用于响应状态数据请求；状态数据权限管理子单元；用于管理状态数据获取权限。

状态数据存储单元，用于状态数据的存储。

状态数据同步单元，用于状态数据的同步。

时间处理子模块，用户获取时间戳，且验证所述时间戳。所述时间处理子模块包括时间定义单元、时间解释单元、时间验证单元、时间标记单元、时间获取单元、时间同步单元。时间戳用于标记某种行为发生的时间，本申请中所述时间戳并不是人类通常所使用的年月日

时分秒时间表示，也不是由共识系统之外的时间源所提供的时间表示，而是以共识系统内部的某种递增数量为基础所形成的时间表示。例如：当本共识节点发生行为 A 时，时间戳获取模块获取到已共识事务的数量，假设数量为 101，则共识时间戳为 101，并为行为 A 打上共识时间戳 101；当共识节点获取到一个已打了共识时间戳（假设为 309）的行为 B 数据时，时间戳验证模块获取共识时间为 101，则由于 309 远大于 101，时间戳验证模块判定行为 B 的时间戳为假，并触发相关处理；当待共识事务 C 的时间戳为 20，而时间戳验证模块获取已共识事务的数量为 101，由于 20 远小于 101 且 C 为待共识事务，则时间戳验证模块判定事务 C 的时间戳为假，并触发相关处理。其中，

时间定义单元，用于存储时间的定义规则，例如以什么部件的状态变化轨迹作为时间参考系；时间解释单元，用于将时间参考系的状态解释为当前时间；时间验证单元，用于验证所获取的时间戳是否正确或有效；时间标记单元，用于对数据添加时间戳；时间获取单元，用于获取当前时间；时间同步单元，用于同步时间；

如图 4 所示，在一种实施方式中，所述系统包括若干个事务输入模块、一个共识网络、若干个状态获取模块。其中，所述共识网络包括 N 个共识节点，所述共识节点包括事务处理、到达序信息、还原序信息、共识序信息、状态数据、时间信息，图 2 是图 4 的一种具体体现，本申请不仅仅包含图 2 一种形式，也包括其他可以体现图 4 的其他方式。通过事务处理子模块、到达绳处理子模块、还原绳处理子模块、共识绳处理子模块、状态处理子模块、时间处理子模块对共识节点做出具体的形式体现，本申请不仅仅包含图 2 一种体现形式。

在一种具体的实施方式中：事务输入模块先后向共识节点 1 发送事务 T0，向共识节点 3 发送事务 T1，向共识节点 2 发送事务 T2，向共识节点 1 发送事务 T3，于是本真序为：T0<T1<T2<T3；各共识节点在获取到事务输入模块直接输入的事务后立即向其他共识节点同步所获取的事务；各个共识节点记录事务到达本共识节点的时序，并同步所述本共识节点到达序，其中：事务到到达序？达共识节点 1 的时序为 T0<T2<T1<T3，事务到达共识节点 2 的时序为：T0<T2<T3<T1，事务到达共识节点 3 的时序为：T1<T0<T3<T2；获取并同步其他共识节点到达序，分析到达序中包含的本真序信息，根据共识节点 1 的到达序明确知道 T0<T3、T2<T3、T1<T3，根据共识节点 2 的到达序明确知道 T0<T2，根据所有共识节点到达序统计发现 T0 更早到达的共识节点数比 T1 的多，T1 更早到达的共识节点数比 T2 多，于是认为 T0<T1、T1<T2，综合所有信息，共识节点 1 还原出时序还原序？为 T0<T1<T2<T3，共识节点 2 还原出时序为 T0<T1<T2<T3，共识节点 3 还原出时序为 T0<T1<T2<T3；共识节点之间共享还原序，最终达成共识序为：T0<T1<T2<T3；执行共识序中的事务，变更状态数据；将共识序中的已共识事务的个数作为一种时间，由于当前共识序为 T0<T1<T2<T3，已共识事务个数为 4，则当前时刻为 4，当有新的事务到达共识节点时，共识节点可以获取当前时刻并为所到达的事务添加时间戳；

状态获取模块向共识节点 3 请求获取状态数据；

在上述具体实施方式中，如果各个共识节点相互信任，并且每个共识节点都诚实地同步自己获取的事务到达序信息，那么共识系统中最终可以达成共识序。但是现实中可能有非诚实共识节点将篡改了的事务到达序同步给其他共识节点，从而影响共识序的形成。例如，共识节点 1 记录的事务到达序为 T0<T2<T1<T3，但是同步给其他共识节点的到达序为 T2<T1<T0<T3。因此，需要有一种方法能避免非诚实共识节点的篡改行为对共识序的影响，也就是每个共识节点必须提供可举证的到达序信息，或者其他共识节点能有效识别非诚实共识节点的篡改行为。

如图 5 所示，所述事务输入模块包括事务发起子模块、事务入口节点分配子模块，事务发送子模块。其中：

事务发起子模块，用于获取事务触发信息，可以通过获取主体的意志输入，完成事务发起，将所述事务提交给事务入口节点分配子模块。事务入口节点分配子模块，用于从共识网络中获取共识节点信息，根据预设入口节点分配规则、待共识事务和所述共识节点，确定所述待共识事务的入口节点；将所述事务提交给事务发送子模块。事务发送子模块，用于将待共识事务传递给入口节点。

如图 6 所示，所述状态获取模块包括状态呈现子模块、状态请求子模块、状态订阅子模块、状态合成子模块、状态存储子模块、状态分析子模块。

状态呈现子模块，显示状态获取模块所获取的当前或历史状态数据。其中，为了离线利用状态数据，减少数据请求次数，状态呈现模块可以暂存已获取的状态数据，而不必每次都向共识节点获取状态数据。例如在股票交易应用中，某只股票的走势图页面就相当于状态呈现模块，这里显示了股票的最新股价和历史价格走势，状态呈现模块可以暂存历史价格数据用于数据分析处理。本申请对已获取的状态数据的存储、分析处理方式不做具体限定。

状态请求子模块，以主动请求获取的方式从共识网络中获取状态数据。例如，在定时刷新页面，每次刷新都是发送一次状态数据请求。状态请求可以是临时发起，也可以是按照预设的规则发起，可以向某一节点发起，也可以同时向多个节点发起，可以按照某种条件一次请求一个或多个状态数据。本申请对状态请求的方式不做具体限定。

状态订阅子模块，以订阅获取的方式从共识网络中获取状态数据。例如，在无人机群应用中，无人机订阅了飞行目标状态数据，当指挥部通过输入事务修改了无人机的预定飞行目标状态数据后，共识网络立即将最新的状态数据推送给无人机。状态订阅可以就某一个或多个状态数据向某一个或多个共识节点发起订阅或终止订阅。

状态合成子模块，将从一个或多个共识节点获取的状态数据做合成处理。例如：由于本

申请所述共识系统中，各共识节点的共识进度会有细微差别，状态数据获取模块从单一共识节点所获取的状态数据可能不是最新数据、可能由于节点故障无法获取数据，为保证获取到最新数据以及保证获取能力，状态数据获取模块可能同时从多个共识节点中获取状态数据，再由状态合成模块从获取的多个状态数据中选择可接受的状态数据，发送给状态数据呈现模块进行处理。

状态存储子模块，用于存储已获取的状态数据。

状态分析子模块，用于分析已获取的状态数据并输出分析结果，所述分析结果的输出还可以经过一系列处理后形成主体的意志并触发事务输入模块 01 进行事务的发起。

实施例 2

本申请提供一种共识节点，应用于所述第一部分的共识网络，如图 7 所示，其中将应用于图 7 的共识节点的相应设备如下，所述共识节点设备包括入口节点设备、到达序节点设备、还原序节点设备、共识序节点设备、状态节点设备、时间节点设备；

所述入口节点设备与所述到达序节点设备、所述状态节点设备数据连接；所述到达序节点设备与所述还原序节点设备数据连接；所述还原序节点设备与所述共识序节点设备数据连接；所述共识序节点设备与所述状态节点设备数据连接；

所述时间节点设备与所述入口节点设备、所述到达序节点设备、所述还原序节点设备、所述共识序节点设备、所述状态节点设备数据连接；

全节点设备具有入口节点、到达序节点、还原序节点、共识序节点、状态节点和时间节点的功能。

其具体的实施过程，发起事务，按照预设分配规则为所述事务分配入口节点，并向所述入口节点输入所述事务；入口节点获取并同步所述事务，对所述事务做一系列处理，将所述事务发送给到达序节点，其中，入口节点根据所述事务携带的入口节点信息以区分事务；到达序节点获取所述入口节点的事务，按照预设到达规则记录本节点到达序，并将所述本节点到达序发送给还原序节点；还原序节点获取所述到达序节点的到达序，按照预设还原规则将所述到达序处理为本节点还原序，并将所述还原序发送给共识序节点；共识序节点获取并同步所述还原序节点的还原序，按照预设共识规则将所述还原序处理为共识序，并将所述共识序发送给状态节点；状态节点根据所述共识序获取和执行所述共识序涉及的事务，更新所述状态数据副本；时间节点利用所述共识序定义一种时间数据，并输入、输出、处理所述时间数据；状态获取模块从状态节点获取状态数据。

与第一方面不同的是：所述共识网络中的共识节点因所负责处理的事务定序共识流程的环节不同而具备一个或多种节点身份。所述节点身份包括：

入口节点，由事务输入模块向入口节点输入事务，客观上产生了“本真序”。入口节点的事务处理子模块按照预分配设规则接收所述事务，并将所述事务输入给到达序节点和状态节点。入口节点设备，包括：入口节点，所述入口节点包括，本地事务处理模块，其中，所述本地事务处理模块获取并同步所述事务，对所述事务做一系列处理，将所述事务发送给到达序节点，其中，所述本地事务处理模块根据所述事务携带的入口节点信息以区分事务。

到达序节点的事务处理子模块接收来自入口节点的事务，客观上产生了“到达序”，由到达绳处理子模块构建到达绳记录到达序信息，并将到达绳输入给还原序节点。达序节点设备，包括：到达序节点，所述到达序节点包括，外地事务处理模块，本地到达绳处理模块，其中，所述到达序节点为所述系统中的所述到达序节点，其中，所述外地事务处理模块，获取所述入口节点的事务，所述本地到达绳处理模块，按照预设到达规则记录本节点到达序，并将所述本节点到达序发送给还原序节点。

优选地，入口节点和到达序节点可以合并为一个节点，从而使入口节点的本地事务不经过网络传输而立即被记录在到达节点的到达绳中，同理，所述入口节点设备和所述到达序节点设备可以合并为一个设备。

还原序节点接收来自到达序节点的到达绳，由还原绳处理子模块处理到达绳，构建还原绳记录还原序信息，并将还原绳输入给共识序节点。还原序节点设备，包括：还原序节点，所述还原序节点包括，外地到达绳处理模块，本地还原绳处理模块，其中，所述还原序节点为所述系统中的所述还原序节点，其中，所述外地到达绳处理模块，获取所述到达序节点的到达序，所述本地还原绳处理模块，按照预设还原规则将所述到达序处理为本节点还原序，并将所述还原序发送给共识序节点。

共识序节点的共识绳处理子模块接收来自还原序节点的还原绳，共识序节点之间达成共识序，由共识绳处理子模块构建共识绳保存共识序信息，并将共识序信息发送给状态节点和时间节点。共识序节点设备，包括：共识序节点，所述共识序节点包括，外地还原绳处理模块，本地共识绳处理模块，其中，所述共识序节点为所述系统中的所述共识序节点，其中，所述外地还原绳处理模块，获取并同步所述还原序节点的还原序，所述本地共识绳处理模块，按照预设共识规则将所述还原序处理为共识序，并将所述共识序发送给状态节点。

状态节点的状态处理子模块获取来自入口节点的事务和来自共识序节点的共识序并按照共识序执行事务，完成对状态数据的变更。状态节点设备，状态节点，所述状态节点包括，外地事务处理模块，状态处理模块，其中，所述状态节点为所述系统中的所述时间节点，其中，所述外地事务处理模块，根据所述共识序获取所述共识序涉及的事务，所述状态处理模块，根据所述共识序执行所述共识序涉及的事务，更新所述状态数据副本。

时间节点由共识序节点向时间节点的时间处理子模块输入所述共识序节点的共识绳状态信息，时间节点将所述共识绳状态信息处理为时间信息，并向入口节点、到达序节点、还原序节点、共识序节点、状态节点、时间节点授时，优选的，入口节点受时并为所述事务添加时间戳，到达序节点受时并为到达绳的当前结添加时间戳，还原序节点受时并为所输出的还原绳添加时间戳，共识序节点为共识绳的当前结添加时间戳，状态节点受时并为状态数据添加时间戳，时间节点受时并与本节点时间进行比对。时间节点设备，包括：时间节点，所述时间节点包括，时间处理模块，其中，所述时间节点为所述系统中的所述状态节点，其中，所述时间处理模块，利用所述共识序定义一种时间数据，并输入、输出、处理所述时间数据。

全节点设备至少包括事务处理子模块、到达绳处理子模块、还原绳处理子模块、共识绳处理子模块、状态处理子模块、时间处理子模块。极端情况下，共识网络中全部为全节点，或者全部都不是全节点。一种全节点设备，是一种共识节点，全节点包括，事务处理子模块、到达绳处理子模块、还原绳处理子模块、共识绳处理子模块、状态处理子模块、时间处理子模块，其中，所述全节点为所述系统中的所述共识节点，其中，事务处理子模块获取并同步所述事务，对所述事务做一系列处理，其中，共识节点根据所述事务携带的入口节点信息以区分事务；到达绳处理子模块记录本节点到达序，并同步所述本节点到达序；到达绳处理子模块获取并同步其他节点到达序，还原绳处理子模块将所述本节点到达序和其他节点到达序处理为本节点还原序，同步所述本节点还原序；还原绳处理子模块获取并同步其他节点还原序，共识绳处理子模块将所述本节点还原序和其他节点还原序处理为共识序；状态处理子模块根据所述共识序获取和执行所述共识序涉及的事务，更新所述状态数据副本；时间处理子模块利用所述共识序定义一种时间数据，并输入、输出、处理所述时间数据。

在一种实施方式中，一个事务定序共识系统中，包括 N 个事务输入模块、F 个状态获取模块和 1 个共识网络，所述共识网络中包括：M 个入口节点、H 个到达序节点、K 个还原序节点、J 个共识序节点、L 个状态节点和 T 个时间节点。由于一个节点可能由多种身份甚至是全节点身份？，所以所述共识网络中的节点总数小于等于（M+H+K+J+L+T），极端情况下，共识网络中全部为全节点，即共识网络中的节点总数为 M，且 M=H=K=J=L=T。

由于共识绳已经决定了事务的执行顺序，接下来每个共识节点各自执行事务实际上是一种重复计算，同时由于共识节点执行效率不同可能会导致各个共识绳的长度不能严格同步，也导致各个共识节点的状态数据不能实时一致。因而可以设立专门的节点负责事务执行和状态数据处理，而共识节点则负责完成事务的共识。

状态节点从共识节点获取最新的未执行共识绳段和相关事务，完成事务的执行，以及状态数据的存储、更新、订阅服务等工作。状态节点与多个共识节点之间可以建立长连接，也可以是自定义的通信方式，共识节点可以监督执行节点，执行节点之间也可以相互监督。

多个共识节点可以共享同一个状态节点，从而可以减轻共识节点的计算和存储压力，也避免了计算和存储资源的浪费。不诚实的执行节点的作恶行为，例如篡改状态数据、不按顺序执行事务等，会很容易被发现，而发生故障的状态节点也容易数据恢复。

实施例 3

本申请提供一种事务定序共识方法，应用于第一方面所述的一种事务定序共识系统，如图 8 所示，首先通过事务输入模块完成以下步骤：

S0001、获取事务触发信息；可以通过获取主体的意志输入，完成事务发起，将所述事务提交给事务入口节点分配子模块。

S0002、获取共识网络中共识节点信息；

S0003、根据所述事务触发信息和共识节点信息，确定待共识事务的事务入口节点信息。

例如，在订票应用中，订票页面就相当于事务发起模块，用户 A 在订票页面中选择影院、电影、场次和座位，后点击“提交”按钮，也就是完成了主体意志的输入，订票页面获取主体的意志输入，完成事务的创建并发起了一个订票事务。进一步地，事务可以由人工触发输入，也可以由设备自主输入，对于事务的发起主体不做具体限定。

一般的，为了促进主体的意志能快速发挥效果，事务发起子模块在获取主体的意志输入后应立即创建事务并提交。但是有时，事务提交的时机由主体控制，事务创建后暂存在事务发起模块，再有主体触发事务发起子模块提交事务。

如图 9 所示，事务入口节点分配子模块获取共识网络中的共识节点信息，包括节点地址、端口和通信协议，可选的，事务入口节点分配子模块获取共识节点的设备运行参数，例如 CPU 空闲率、内存使用率；事务入口节点分配子模块从事务发起子模块中获取事务；事务入口节点分配子模块按照预设的入口节点分配规则根据节点的信息，确定所述事务的入口节点。其中，所述入口节点分配规则可以依照指定、轮询、随机、就近、负载均衡或其他原则进行预设。例如：方式 1：按照指定原则，事务入口节点分配子模块指定节点 0 为事务 1 的入口节点；方式 2：按照轮询原则，假设一共 N 个节点，事务入口节点分配子模块为第一个事务指定第一个节点，为第二个事务指定第二个节点，为第 N+1 个事务指定第一个节点；方式 3：按照随机原则，事务入口节点分配子模块随机指定入口节点；方式 4：按照就近原则，事务入口节点分配子模块已知与各节点通信延迟信息的前提下，选择通信延迟小于某个阈值的节点之一作为事务的入口节点；方式 5：按照负载均衡原则，事务入口节点分配子模块已知与各节点运行负载信息的前提下，选择运行负载小于某个阈值的节点之一作为事务的入口节点；方式 6：按照其他原则，例如，事务入口节点分配子模块将事务 ID 尾号为 5 的事务分配给节点 ID 尾号也为 5 的节点，事务入口节点分配子模块将事务 ID 尾号为 6 的事务分配给节点 ID 尾号也

为 6 的节点，依此类推。

然后，所述一种事务定序共识方法，如图 10 和图 11 所示，包括如下步骤：

S1001、获取待共识事务和相应的事务信息，所述事务信息包括事务编号、事务输入模块信息、事务入口节点信息，事务待执行信息和获取待共识事务的路径信息；所述事务入口节点信息通过预设分配规则为所述待共识事务分配的入口节点，其中，所述预设分配规则包括指定、轮询、随机、就近和负载均衡；根据所述事务入口节点信息区分所述待共识事务。其中，如图 12 所示，所述待共识事务，包括但不限于：事务 ID、事务输入模块 ID、入口共识节点 ID、事务待执行内容。所述事务输入模块可能是用户客户端，也可能是其他设备，这里不做具体限定。系统通过密码学手段实现对事务内容是否被篡改的可验证性。其中，事务待执行内容中一般包括：主体 ID、指定的事务执行器 ID、待执行状态更新描述指令。

S1002、根据所述获取待共识事务的路径信息，将所述待共识事务划分为本地事务和外地事务；其中，本地事务。是对于某个共识节点来说，以本共识节点作为入口节点进入网络并被本共识节点获取的事务。在本申请中，有时指一个事务，有时指一组事务，根据语境而定。外地事务。是对于某个共识节点来说，以其他共识节点作为入口节点进入网络并被同步到本共识节点的事务，在本申请中，有时指一个事务，有时指一组事务，根据语境而定。其中，所述获取待共识事务的路径信息通过获取事务同步消息中的路径表；将本节点的 ID 添加到所述事务同步消息的路径表的末尾获取。

S1003、验证、解析、同步所述待共识事务；其中，如图 13 所示，对于验证、解析所述待共识事务方法，包括：

S2001、根据预设验证项对所述待共识事务进行验证，得到验证结果；S2002、当验证结果通过时，将所述待共识事务标记为已验证待共识事务；S2002、根据预设解析规则对所述已验证待共识事务进行解析，得到解析结果；S2002、将所述已验证待共识事务标记为已解析待共识事务。

其中，在 S2001 中，如图 14 所示，验证本地事务的方法的示意图。所述方法的执行载体为本地事务验证子单元。所述方法的具体步骤为：本地事务验证子单元从本地事务获取子单元获取到一个本地事务；本地事务验证子单元对所述待验证事务验证项进行逐项验证，例如，包括下列验证项的部分或全部，但不限于下列验证项：

（1）检查所述待验证事务是否已经为重复接收。例如，获取到事务 ID 后，从已获取的事务中查找是否包括有所述 ID，如果有，则说明为重复接收，则验证不通过；

（2）检查所述待验证事务签名是否正确。例如，按照数字签名的验证方法，发现待验证事务的数字签名与密钥不匹配，则判定本项验证不通过；

（3）检查所述待验证事务的结构是否符合预设规定结构。例如：预设规则中规定了事务应当以键值对格式表达，而待验证事务没有按照键值对格式表达，或者键名与规定不对应，或者键值对的排列顺序不符，则判定本项验证不通过；

（4）检查所述待验证事务的组成内容是否符合预设规定结构。例如，预设规则中规定了事务 ID 为 20 位小写字母，而待验证事务的事务 ID 为 19 位，则判定本项验证不通过。例如，待验证事务的待执行内容为空，则判定本项验证不通过。

只有全部验证通过才判定验证通过，否则判定验证不通过；上述验证项也可以并行验证。

本地事务验证子单元根据验证结果，完成处理，即，如果验证通过，则调用事务标记单元将所述待共识事务标记为“已验证”，并将所述事务提交给本地事务解析子单元；如果验证不通过，则处理所述事务，例如，将所述事务从事务存储单元中删除，或者调用事务标记单元将所述事务标记为“验证不通过”，并可选的，将所涉及的事务输入模块标记为“不可信”；

如图 15 所示，验证外地事务的方法。具体步骤为：外地事务验证子单元获取到一个待验证的外地事务同步消息；外地事务验证子单元对待验证的外地事务同步消息进行验证，例如，包括下列部分或全部验证事项但不限于下列事项：

（1）检查所述待验证事务是否为重复接收；

（2）检查所述待验证事务的签名是否正确；

（3）检查所述待验证事务的时间戳是否有效。例如：设定正确的时间戳结构为（时间数字，时间戳校验码），获取到本地当前共识时间戳并估计待验证事务的时间数字误差范围，而待验证事务的时间戳结构不对应，或者时间数字与时间戳校验码不对应，或者时间数字不在所述预设误差范围内（远大于或远小于）；

（4）检查所述待验证事务的结构是否正确；

（5）检查所述待验证事务的内容是否正确；

只有全部验证通过才判定验证通过，否则判定验证不通过；上述验证项也可以并行验证；

外地事务验证子单元根据验证结果，完成处理，即，如果验证通过，则调用事务标记单元将所述待共识事务标记为“已验证”，并将所述事务提交给外地事务解析子单元；如果验证不通过，则处理所述事务，例如，将所述事务从事务存储单元中删除，或者调用事务标记单元将所述事务标记为“验证不通过”，并可选的，将所涉及的事务输入模块标记为“不可信”。

在步骤 S2002 中，解析本地事务的方法。所述方法的具体步骤为：本地事务解析子单元获取到一个已验证事务；本地事务解析子单元按照预定解析规则获取到解析结果，为所述事务中的信息，一般包括：事务输入模块 ID、事务入口共识节点 ID、事务 ID、事务待执行内

容；本地事务解析子单元将解析结果存储至到事务存储单元；本地事务解析子单元调用事务标记单元将所述事务标记为“已解析”。解析外地事务的方法。所述方法的具体步骤为：外地事务解析子单元获取到一个已验证的外地事务；外地事务解析子单元处理外地事务获取到解析结果，一般包括：事务输入模块 ID、事务入口共识节点 ID、事务 ID、事务待执行内容；外地事务解析子单元将外地事务解析结果存储在事务存储单元中，并将所述外地事务标记为“已解析”。

如图 16 所示，共识节点存储本地事务的方法。所述方法的具体步骤为：事务存储单元获取空闲的存储空间；事务存储单元获取来自本地事务获取子单元的本地事务存储请求或来自外地事务获取子单元的外地事务存储请求，处理所述本地事务存储请求或外地事务存储请求，存储所述本地事务和外地事务；事务存储单元获取来自事务标记单元的本地或外地事务标记请求，处理所述本地或外地事务标记请求，将所述事务标记为指定值；事务存储单元获取本地或外地事务删除请求，处理所述事务删除请求，将所述事务从事务存储模块删除；事务存储单元获取本地或外地事务查询读取请求，查询事务，并输出查询结果；所述事务存储单元满足事务的存储管理需求，本质上是一个数据库，具体存储方式不做限定，例如：可以是把事务存储在内存中，也可以把事务存储在磁盘文件中，也可以使用关系数据库、对象数据库、图数据库等存储。

如图 17 所示，共识节点获取外地事务的方法。所述方法的具体步骤为：外地事务获取子单元获取来自其他共识节点的事务同步消息；外地事务获取子单元从所述事务同步消息中获取待共识事务；外地事务获取子单元将所述事务提交给事务存储单元进行存储；外地事务获取子单元将所述事务提交给外地事务验证子模块。

其中，对于同步所述待共识事务的方法，如图 18 所示，包括：S3001、获取所述已验证待共识事务及当前共识时间戳；S3002、创建待共识事务同步消息，所述待共识事务同步消息包括待同步事务、消息发布时间戳、共识节点签名；S3003、根据网络拓扑数据，按预设生成原则生成已验证待共识事务同步算法；S3004、根据所述已验证待共识事务同步算法对所述已验证待共识事务进行同步，得到已同步待共识事务。

本地到达绳同步触发条件时向其他共识节点同步所述本地到达绳中未同步绳段的方法。所述方法的具体步骤为：判断本地到达绳当前是否触发了本地到达绳同步条件，如果是则执行下一步骤，否则等待本地到达绳下一次打结完成后再执行本步骤；生成本地到达绳同步消息，获取本地到达绳的同步策略，按照同步策略进行同步。可选的，被写入本地到达绳同步消息的是压缩后的本地到达绳段；将所述被同步的本地到达绳段标记为已同步。例如：可以设置一个整数变量，记录已同步的本地到达绳段的末尾绳结的编号，表示所述编号绳结及以前的绳结已被同步。所述本地到达绳同步消息，至少包括：消息发布时间戳，所述时间戳为

消息发布时共识绳中的末尾绳结的序号和哈希值；待同步本地到达绳，可以是已压缩的；消息的发布者共识节点的签名。所述同步算法可以是 PBFT、雪崩、Gossip 等，也可以是共识节点自主约定的算法，本申请不做具体限定。

同步外地事务的方法。所述方法的具体步骤为：

方式一：外地事务同步子单元获取到一个已验证的外地事务；外地事务同步子单元从时间处理子模块获取当前的共识时间生成共识时间戳，生成外地事务同步消息，所述同步消息一般包括所述外地事务、当前共识时间戳、本共识节点签名；外地事务同步子单元，根据共识网络的网络拓扑数据生成外地事务同步算法，或者按照预设的同步策略作为所述外地事务的同步算法；外地事务同步子单元将所述外地事务同步消息按照所述外地事务同步算法发送给共识网络的其他共识节点，待所述外地事务同步消息发出后，调用事务标记单元将所述外地事务标记为“已同步”。

方式二：外地事务不经过验证而直接被本共识节点同步给其他节点，其目的避免因验证外地事务和生成新的外地事务同步消息而造成的延迟。所述外地事务同步消息，至少包括：消息发布时间戳，所述时间戳为消息发布时共识绳中的末尾绳结的序号和哈希值；待同步外地事务；消息的发布者共识节点的签名。所述同步算法可以是 PBFT、雪崩、Gossip 等，也可以是共识节点自主约定的算法，本申请不做具体限定。

S1004、获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳。在进行 S1005 步骤中，同步不同共识节点之间的本地到达绳之前，包括：对所述本地到达绳进行分段处理，得到到达绳段；根据预设压缩规则对所述到达绳段的进行压缩处理；所述压缩处理保留所述到达绳段上的事务结，并是删除所述到达绳段的非标记用空结。

下面具体介绍到达绳的目的和作用：如果一个共识节点可以尽快地将所接收到的事务引入当前结，则所述当前结生成的时间近似等于该事务被共识节点接收的时间。一般的，打结频率越高，则所述时间近似误差越小。于是，先被该共识节点打入绳结的事务可以视作先到达该共识节点或者说先被该共识节点接受。于是，所述结绳上蕴含了事务到达本共识节点相对时序的证据信息。由于所述结绳记录了事务到达共识节点的时序证据，故称作所述结绳为“到达绳”。所述到达绳相比于比特币方案的交易缓冲区，至少增加了事务到达共识节点、事务到达共识节点的时间、事务输入模块三个信息维度，可以极大地扩展了信息表达能力。到达绳不仅能反映事务到达共识节点先后时序，还能一定程度上反映事务到达共识节点的频率变化，进一步还能反映当前网络拥堵情况，还能近似反映不同共识节点的相对打结频率，还能反映出本地事务晚于此前到达的外地事务进入网络。从安全性角度讲，由于结绳的每个绳结都是前向依赖的，如果想逆转两个事务的顺序就必须重新花费时间构造所涉及的绳段，而在构造的过程中又需要把不断新接收的事务引入结绳，否则会积压多个待引入的事务而不得

不连续地引入事务，这种伪造行为会在结绳上留下痕迹并且很容易被其他节点发现。然而，单一节点对事务到达时序的伪造难以影响到整个网络最终对事务进入网络时序的共识。可见，节点伪造到达绳是徒劳的。到达绳可以视作一种时序数据流，通过选择不同段的到达绳，利用统计和概率知识可以分析事务到达频率的变化情况，分析发现不同事务发送主体的事务到达规律，分析事务发送主体的行为动机，分析事务背后的宏观状态，分析预测未来某个时间段的事务到达频率，而所述这些分析的结果可以为共识系统、设备、应用的演化，以及为用户的决策提供依据。到达绳的分析应用所涉及的方法均是容易想到的，也在本申请保护范围内。

初始化本地到达绳的方法具体步骤为：所述方法的目的在于确定本地到达绳的初始绳结，由于共识节点可能采用同一个数字身份而多次进入或退出共识网络，每次退出共识网络后本地到达绳就会停止打结从而造成间断，因此每次进入共识网络都要告知共识网络即将初始化新的本地到达绳，并需要确定本次的初始绳结。流程包括：在有 N 个共识节点的共识网络中，1 号共识节点加入共识网络后，向共识网络的各共识节点发布一个特定类型的“本地到达绳初始化事务”，所述事务至少包括：本共识节点签名、本共识节点上次加入共识网络产生的本地共识绳的末尾绳结，如果是第一次进入网络，则注明是第一次加入，且末尾绳结为空或特定值；共识网络对所述事务达成共识，共识节点将所述事务写入共识绳并按照预设方法对所述事务进行执行，执行结果是，根据所述事务数据和所述事务所在的共识结绳数据生成一种选择策略，所述选择策略通过选择出一组共识节点，例如：2、3、4 号节点来回复对 1 号共识节点的所述事务的验证结果；以所述 2 号节点为例，收到所述消息后，检查本地是否曾经接收过 1 号节点发来的 1 号节点本地到达绳，并检查与 1 号节点发来的所述消息内容是否一致，即，如果本地曾经接收过 1 号节点本地到达绳，则获取到最近一次接收的所述 1 号本地到达绳的末尾绳结，并检查是否与所述来自 1 号节点的本地到达绳初始化消息中的所述末尾结绳数据是否一致；如果本地不曾接收过 1 号节点本地到达绳，则检查所述消息是否表达共识节点第一次进入网络。如果检查通过，则向 1 号节点回复消息，所述消息表达“已接受 1 号节点的本地到达绳初始化请求”，并附带所述事务在共识绳中所在的绳结，记作 0 号绳结；1 号共识节点接收来自 2、3、4 号节点的回复，如果 50%以上回复为“已接受”，确认所述事务已经被共识网络完成共识且被接受，则本地到达绳初始化完成，并将所述 0 号绳结作为 1 号节点本地到达绳的初始绳结所要引用的数据之一。

所述方法的一种特殊情况是，在共识网络最初组网的时候，最初的几个共识节点的本地到达绳的初始绳结所要引用的数据为预设值，共识节点各自生成本地到达绳的初始绳结后开始各自打结延长本地到达绳，每个共识节点向共识网络中输入“本地到达绳初始化事务”触发共识网络的共识绳初始化。

所述方法另一种方式是：共识节点向其他共识节点询问本共识节点是否曾经向其发送过本地到达绳数据，如果大多数或全部共识节点返回为没有接收过所述到达绳数据，则说明本地共识节点需要从零开始生成本地到达绳，将剩余的共识节点标记为可疑节点；如果大多数或全部节点返回了所述到达绳数据的最新绳段且所述绳段中最新的绳结相同，则说明本地到达绳曾经生成过本地到达绳，则需要接续之前的本地到达绳继续生成，将所述最新的绳结作为本次生成到达绳的初始绳结，将剩余的共识节点标记为可疑节点。

S1005、同步不同共识节点之间的本地到达绳，得到同步后各节点到达序信息；上述 S1004 和 S1005 中，具体方法，如图 19 所示，包括：S4001、初始化共识节点的本地到达绳；S4002、不间断地创建绳结且延长所述本地到达绳，所述绳结包括绳结编号、引入绳结数据和校验项；S4003、当所述共识节点接收到本地事务时，将所述本地事务引入所述本地到达绳的当前绳结；S4004、当在满足预设的本地到达绳的同步触发条件时，向本节点外共识节点同步所述本地到达绳中未同步部分；S4005、根据预设到达规则接收、验证和同步外地到达绳；S4006、解析所述本地到达绳和所述外地到达绳，得到同步后各节点到达序信息。

其中，不间断地打结延长所述本地到达绳，并将到达本节点的本地事务或外地事务立即引用到当前绳结中的方法。所述方法的具体步骤为：本地到达绳同步模块确认本地到达绳的初始绳结所要引用的数据后，有本地到达绳打结模块打结本地到达绳的初始绳结，设置初始结绳编号为 0，此时，所述初始绳结也是所述本地到达绳的末尾绳结；本地到达绳打结模块立即获取到本地到达绳的末尾绳结，并立即计算出所述末尾绳结数据的数字摘要；本地到达绳打结模块立即检查是否有未被引入本地到达绳的待共识事务，如果有则锁定其中一个事务，获取所述事务的数字摘要，并确定当前结为事务结，而如果没有未被引入的待共识事务，则确定当前结为空结；创建当前结，所述当前结的编号为所述末尾绳结的编号加 1，并将所述末尾绳结作为当前结的内容之一，如果当前结为事务结，则将所述事务的数字摘要作为当前结的内容之一，而如果当前结为空结，则将默认值作为当前结内容。可选的，统计从初始节点到本节点的事务结的个数，作为当前结的内容之一。可选的，获取当前结之前的上一个事务结的编号，作为当前的内容之一。所述两个可选项可以增加到达绳的重建难度，增加对到达绳数据的验证方法，不过也会增大到达绳的数据量。将所创建的当前结存储在本地到达绳数据中。触发下一次打结流程，直到共识节点按照预设原因停止本地到达绳打结。

例如，在 1 号节点中，本地到达绳不间断打结，在 300 号绳结打完之后，待共识事务 1 到达了 1 号节点，同时，本地到达绳开始打下一个结，此时获取到末尾绳结为 300 号绳结，则当前绳结的编号为 300+1=301，计算出 300 号绳结的哈希值 h300，检查到有未被引入的待共识事务 1，计算出事务 1 的哈希值 hT1，将 h300 和 hT1 作为被引入的数据，打结完成第 301 号绳结，301 号绳结为事务结。立即开始打下一个结，末尾绳结为 301 号，则当前绳结编号为

301+1=302，未发现有待引入的事务，则第 302 号绳结打为空结。此时几乎同时到达了事务 2 和 3，则第 303 号绳结引入事务 2 的哈希值和第 302 号绳结的哈希值，打为事务结，而第 304 号绳结引入事务 3 的哈希值和第 303 号绳结的哈希值，打为事务结。

其中，采用计算 hash 值（计算数字摘要）的方式的目的是利用一种在一定约束条件下不可逆的计算方式来标记一段时间延迟，而且这种时间流逝是可验证的。如果采用了其他可验证时间延迟的方法来打结，应当认为是容易想到的，也在本申请的保护范围内。

其中，如图 20 所示，到达绳的结构由一组连续的前向依赖数据区块组成，所述前向依赖数据区块是指一般的，第 N+1 数据区块的创建依赖于第 N 数据区块的数据作为输入。有前向依赖关系的数据区块必然是串行创建，于是可以利用数据区块的创建来锚定一段时间的流逝，每个数据区块的创建都意味着时间的推进。其中，所述构造前向依赖的数据区块的过程称作“打结”；所述前向依赖的数据区块称作“绳结”；所述连续打结所形成的结集合称作“结绳”；所述结绳中的一组连续的绳结称作“一段结绳”或“绳段”。按照上述定义，到达绳的结构包括：一组连续生成的绳结。为了区分到达绳的来源，对于一个共识节点来说，由本共识节点打结而成的到达绳称作“本地到达绳”，接收的来自其他共识节点的到达绳称作“外地到达绳”。

其中，如图 21、图 22 所示，绳结的结构示意图，所述绳结的结构包括：绳结编号，一般为连续的自然数，且在同一个到达绳中后产生的绳结编号要比之前生成的绳结的编号大。例如：第一个绳结的编号为 0，第二个为 1，第 N 个为 N-1；前序结引用，一般为同一个本地到达绳中紧邻本绳结的上一个绳结的数字摘要值。例如，当前的绳结编号为 9，则前序结为 8 号绳结，前序结引用为 8 号绳结的哈希值；事务引用，一般为一个待共识且已验证的事务的数字摘要值，表示引入了所述事务，或者为空值或特定值，表示本绳结没有引入事务；校验项，可选的，为了增加到达绳的重建难度并易于检查被篡改的绳结，可以设置零个或多个校验项，所述校验项可以是当前绳结所见证的一些特征状态值，在一种实施方式中：到当前绳结为止所在的本地到达绳中引入的事务的总数；当前绳结的上一个引入了事务的绳结的编号；当前绳结向前一定数量的绳结中引入的事务的总数；当前绳结引入的事务的入口共识节点的 ID；当前绳结之前第 N 个满足某特征的绳结的数字摘要值；其他预设的特征值。校验规则编号，可选的，为了灵活地在不同绳结中增减或变更校验项，可以设置校验规则的编号，指定每个校验项按照哪个校验规则来校验。从程序的角度来看，所述校验规则为一段可以验证所述校验码真伪的程序，在共识节点中预设了一组校验规则，给每个规则设置一个编号，那么通过绳结中的校验规则编号即可知道执行那段校验规则。

在本申请中，一个绳结可以用一个结构体表示，存储在内存中，也可以用 JSON、XML 等形式存储在文件、数据库、磁盘中，只要有预设格式保存上述编号、前序结引用、事务引用、校验项等内容即可，对绳结的格式和存储位置不做限定。构造当前结时，检查当前是否有未

引入到此前结绳的待共识事务。其中，如果当前有 0 个所述待共识事务，则将代表“空事务”的特定值引入当前结中，该绳结被称作“空结”；如果当前有 1 个所述待共识事务，则将该事务的哈希值引入当前结，该绳结被称作“事务结”；如果当前有 N>1 个所述待共识事务，则选择其中 1 个事务，并将该事务的哈希值引入当前结，以此类推，将剩余的 N-1 个事务依次引入相邻的 N-1 个未来的当前结；优选地，共识节点的打结频率应当大于事务到达的频率，避免出现同时多个事务等待引入的情况。极端情况下，每个结都是事务结，即理论上，在不考虑网络带宽、I/O 瓶颈等其他限制因素的情况下，对待共识事务处理速率的上界等于打结频率。

其中，在 S4002 中，绳结编号为连续得自然数，且于所述绳结的产生时间相对应；将所述本地到达绳作为本节点的计时手段；所述本节点计时手段的当前时刻值为所述本地到达绳的当前的绳结编号。

其中，如图 23 所示，在一种实施方式中，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器和调度器，包括：依次异步调用空闲的打结器进行打结操作；生成绳结编号、确定前序绳结以及待映入绳结的事务，并创建引入绳结。

如图 24 所示，在另一种实施方式中，所述不间断地创建绳结且延长所述本地到达绳，应用于 N 个待执行的打结器，且 N 个打结器各自有唯一的编号并持续竞争一个互斥锁，所述互斥锁携带一个整数变量 a 用于生成绳结编号和一个变量 b 用于记录最近一次竞争到互斥锁的打结器的编号；包括：当一个打结器竞争到互斥锁后，将互斥锁所携带的整数变量 a 加 1 从而生成当前绳结编号；读取互斥锁变量 b 从而获取上一个竞争到互斥锁的打结器的编号；将变量 b 置为本打结器的编号，并判定是否有待引入绳结的事务；当有待引入绳结的事务，则在所述本地到达绳上打事务结；当没有待引入绳结的事务，则立即释放互斥锁并打空结；其中，所述事务结包括本打结器的编号、上一个竞争到互斥锁的打结器的编号；所述空结为没有引入事务且已创建完毕的绳结。

上述两种实施方式中，所述本地到达绳的打结包括：获取本地到达绳中含有当前结的绳段；统计所述含有当前结的绳段中事务结的个数占所述含有当前结的绳段中绳结总数的占比；当所述占比超过预设阈值时，增加本地到达绳的打结器；当所述占比低于预设阈值时，减少本地到达绳的打结器。

其中，在步骤 S4005 中，如图 25 所示，共识节点接收到达绳的方法，具体步骤为：外地到达绳获取子单元获取来自其他共识节点的外地到达绳同步消息；外地事务获取子单元从所述外地到达绳同步消息中获取外地到达绳绳段；外地事务获取子单元将所述外地到达绳绳段提交给外地到达绳存储子单元进行外地到达绳存储；外地事务获取子单元将所述外地到达绳绳段提交给外地到达绳验证子单元，触发外地到达绳验证子单元进行外地到达绳验证；外地

事务获取子单元将所述外地到达绳绳段提交给外地到达绳拼接子单元，触发外地到达绳拼接子单元进行外地到达绳拼接；外地事务获取子单元将所述外地到达绳绳段提交给外地到达绳同步子单元，触发外地到达绳同步子单元进行外地到达绳同步；外地事务获取子单元将所述外地到达绳绳段提交给外地到达绳解析子单元，触发外地到达绳解析子单元进行外地到达绳解析；上述方法中，解析、验证、拼接可以调换次序或者同时完成，即，可以解析和验证可以同时完成，也可以先验证后拼接，也可以先解析后验证，调换次序后的新方法的结果是一样的，即把已验证已拼接完成的绳段的解析结果存储在到达绳解析结果存储模块中。

其中，在步骤 S4005 中，验证外地到达绳的方法包括：获取待验证外地到达绳；根据预设到达绳验证项对所述待验证的外地到达绳段进行验证，得到验证结果；其中，所述预设到达绳验证项包括外地到达绳的签名、外地到达绳中每个绳结、每个绳结对应的事务、每个绳结的校验值。

在一种具体实施方式中，共识节点验证外地到达绳的方法。具体步骤为：外地到达绳验证子单元获取到待验证的外地到达绳段；外地到达绳验证子单元验证所述外地到达绳的签名是否正确，如果验证不通过则停止验证并输出验证结果触发相应流程，如果验证通过则执行下一步骤；外地到达绳验证子单元验证所述外地到达绳中每个绳结是否正确，避免恶意共识节点伪造绳结。例如：（1）获取当前绳结的前序结，按照前序结引用的生成规则生成所述前序结引用值，验证所述引用值与当前绳结的前序结引用值是否相等。（2）获取当前绳结的校验码和校验规则编号，按照校验规则编号获取对应的校验规则，按照所述校验规则校验所述验证码。如果验证不通过则输出验证结果并触发相应流程，如果验证通过则执行下一步骤；外地到达绳验证子单元验证所述外地到达绳中的所有事务是否已接受，避免恶意共识节点通过伪造事务的哈希值而生成无效的绳结；如果验证不通过则输出验证结果并触发相应流程，如果验证通过则执行下一步骤；将验证通过后的所述外地到达绳段标记为已验证。例如：设置一个二元组变量（N，V），N 为所述外地到达绳的源共识节点的编号，V 记录已验证的外地到达绳段的末尾绳结的编号，例如，当 N=1，V=100 时，表示所述来自 1 号共识节点的外地到达绳的 0-100 绳结组成的绳段已经完成验证；对于验证不通过的外地到达绳的源共识节点执行相应的处罚措施。

对所述待验证的外地到达绳段进行验证包括随机验证、并行验证和协作验证。

如图 26 所示，一种共识节点随机验证外地到达绳的方法。具体步骤为：外地到达绳验证子单元随机获取待验证的外地到达绳的一个绳段；外地到达绳验证子单元对所述绳段进行验证，输出验证结果，验证不通过则输出验证结果并触发相应流程，例如：向共识网络发布验证不通过的通知，所述发布通知的方式可以是广播消息，也可以是发布一个待共识事务。进行随机验证可以减轻共识节点的验证压力，但是随机验证可能会避开恶意的绳结，为了避免

这种情况，但是，当多个共识节点都随机验证一个绳段时，绳段的覆盖率就会接近甚至达到完全覆盖，另外，由于随机验证对绳段选择的不确定性，恶意节点伪造到达绳仍然会面临被发现的风险，从而放弃侥幸心理。

如图 27 所示，一种共识节点并行验证外地到达绳的方法。具体步骤为：外地到达绳验证子单元实施一组并行的验证程序，所述每个验证程序都获取所述外地到达绳的一个绳段并输出验证结果为通过或不通过，如果不通过则输出相应的验证不通过的证据，例如指出某个绳结的校验码校验错误，指出某个绳结的；一个程序汇总验证结果，当所有的验证结果都为通过时才视作验证通过，否则验证不通过。如果验证不通过则输出验证结果并触发相应流程。为了支持本方法，外地到达绳在同步之前可以进行定制的压缩，例如：保留事务结和编号为 100 的整数倍的绳结，于是所述外地到达绳在被验证的时候可以每 100 个绳结作为一个绳段，从而被并行验证。

在一种实施方式中，一种一组共识节点协作验证外地到达绳的方法。具体步骤为：一组共识节点的外地到达绳验证子单元完成对待验证外地到达绳段验证任务的分工，分别验证待验证外地到达绳段的部分绳段；所述各外地到达绳验证子单元分别完成验证后相互告知验证结果；各外地到达绳验证子单元分别收集所述验证结果，当所有验证结果均为验证通过后，则确定所述外地到达绳段验证通过，否则，验证不通过；

其中，如图 28 所示，同步外地到达绳的方法，包括：S5001、获取所述外地到达绳；S5002、创建外地到达绳同步消息，所述外地到达绳同步消息包括待同步外地到达绳、消息发布时间戳、共识节点签名；S5003、将所述外地到达绳写入所述外地到达绳同步消息，生成外地到达绳同步算法；S5004、根据所述外地到达绳同步算法向其他共识节点发送外地到达绳同步消息，得到已同步外地到达绳。

如图 29 所示，共识节点存储外地到达绳的方法。所述方法的具体步骤为：外地到达绳存储子单元获取空闲存储空间；外地到达绳存储子单元获取来自外地到达绳获取子单元的外地到达绳段并将所述绳段存储到所述空闲存储空间；外地到达绳存储子单元获取外地到达绳删除请求，对指定的外地到达绳段进行删除；外地到达绳存储子单元获取外地到达绳查询请求，和查询请求条件查询存储空间内的外地到达绳段数据并输出查询结果；外地到达绳存储子单元获取外地到达绳标记请求，将指定的外地到达绳段标记为指定状态；外地到达绳存储子单元本质上是一个存储外地到达绳数据和外地到达绳段标记状态的数据库，完成对外地到达绳段的存储、删除、查询和标记。

S1006、根据预设还原规则分析所述各节点到达序信息，得到本节点中两两事务的还原序以及一组事务连续的还原序信息，构建本地还原绳；其中，两两事务的还原序以及一组事务连续的还原序信息，包括：获取所述到达序信息中涉及的事务信息；在所述到达序信息中涉

及的事务中选择两个事务；判断所述两个事务的还原序，输出所述两个事务的还原序信息和对所述两个事务的还原序信息的置信度；迭代至获得所述到达序信息中一组事务连续的还原序。交换不同共识节点之间的本地还原绳，包括交换还原序信息的共识节点组合：获取共识节点信息，获取到达绳、还原绳和共识绳的作为初始值；根据所述初始值生成随机数；根据共识节点信息和所述随机数产生交换还原序信息的节点组合。

在一种实施方式中，还提出一种共识节点并行地处理还原绳数据两两事务还原序的方法。所述方法的具体步骤为：对于 N 个待共识事务，产生 M=N*(N-1)/2 对事务，也就是 M 个时序分析任务；利用多核资源（例如多 GPU、多 CPU）对 M 个时序分析任务并行处理，可以是每个核分别处理一个任务，也可以是调度多个核共同处理 M 个任务；每得出一对事务的时序分析结果，就立即写入到还原绳。

S1007、交换和同步不同共识节点之间的本地还原绳，得到同步后各节点还原序信息。

S1008、根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，所述共识绳包含已共识事务。在 S1007 和 S1008 的步骤中，如图 30 所示，根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，包括：S6001、获取本节点还原绳中未共识且已稳定的还原绳段；S6002、接收其他 N 个共识节点的对应的请求未共识且已稳定的还原绳段；S6003、根据已获取的到达序信息和还原序信息检查外地还原绳；S6004、将接收的请求未共识且已稳定的还原绳段与本节点对应的共识且已稳定的还原绳段比较，保留不同绳段中相同率大于三分之二的时序关系；S6005、重复与多个请求未共识已稳定的还原绳段比较，直到获得相同的时序关系，得到共识序信息，构建共识绳。

S1009、根据对所述共识绳解析，得到共识绳中已共识事务。

S1010、根据已共识事务更新状态数据。根据已共识事务更新状态数据，如图 31 所示，包括：S7001、按照共识序信息解析共识绳中已共识事务；S7002、根据所述已共识事务创建事务执行队列；S7003、将所述已共识事务写入所述事务执行队列；S7004、按队列处理规则，通过执行器对所述已共识事务执行队列中的事务进行处理。

解析本地共识绳中的事务，创建事务执行队列，将可执行的事务写入所述事务执行队列的方法。所述方法的具体步骤为：共识绳事务解析模块获取到共识绳中未被解析执行的绳段中最早的绳结，根据共识绳结中的事务引用查询到对应的事务，将所述事务 push 进入执行队列，重复执行本步骤直至所述全部结绳中的事务执行任务均被 push 进入执行队列；共识绳事务执行模块按照先进先出的顺序，获取到所述执行队列中的队首的事务 T；共识绳事务执行模块获取事务 T 的待执行内容，按照所述待执行内容指定的执行器 ID 获取执行 T 的执行器 E。其中，所述执行器 E，本质上是一段可以执行事务的程序，可以是一个执行器，也可以是

多个执行器按照一定结构构成的执行器组合；将所述事务 T 的待执行内容载入执行器 E，执行器 E 根据预设的执行规则执行所述待执行内容，从而改变状态数据或不改变状态数据。其中，如果 T 的待执行内容不符合预设的执行条件，则执行器 E 事实上拒绝执行 T，也就是说，事务中携带的是事务发起方对状态数据的变更请求，但是所述请求是否被响应以及如何响应，则要由所述事务对应的执行器来判断和执行；执行器 E 执行完成后，将 T 标记为“已执行”，共识绳事务执行模块将已执行事务移出执行队列。

可选的，可以由另一个并行的程序来执行。例如，将共识绳中的未执行事务按照从早到晚的顺序，把相互之间有排队需求的事务分配在同一个执行队列中，而共识节点按照执行队列先进先出的顺序执行队列内的事务。共识节点可能同时存在多个执行队列，执行队列可以并行执行。

执行器对所述事务执行队列中的事务进行处理，如图 32 所示，包括：S8001、执行器获取所述已共识事务的待执行内容；S8002、检查所述待执行内容；S8003、将所述待执行内容转换为可执行指令；S8004、当对应事务满足可执行指令的执行条件，按照可执行指令变更状态数据。

在 S1010 中更新状态数据之后，所述事务定序共识方法还包括：共识节点根据共识绳的状态变化轨迹得到共识时间。所述共识时间的当前时间包括当前时刻值和所述当前时刻值的校验值；当前时刻值为所述共识序中当前已共识事务的个数；获取所述共识时间的当前时间为时间戳。

本实施例一种事务定序共识方法应用在一种事务定序共识系统的具体工作流程如下：

事务输入模块各自向共识网络输入事务，客观上产生了“本真序”。其中，事务输入模块发起待共识事务，为所述发起事务分配入口节点，并向所述入口节点提交所述事务。其中，事务输入模块可以采用指定、轮询、随机、就近、负载均衡或其他原则确定为事务分配入口节点的方法。

事务处理模块流程。共识节点接收并同步所述事务，客观上产生了“到达序”。其中，共识节点的事务处理子模块接收本地务和外地事务，并验证、解析、存储和同步本地事务和外地事务。优选地，在所述同步事务时标注所述事务从入口节点到达本节点经过的路径信息（如经过的节点列表或节点个数）。

到达绳处理模块流程。共识节点各自以相互可理解的方式记录下到达序信息。其中，共识节点的到达绳处理子模块初始化本地到达绳后不间断地打结延长所述本地到达绳，并将到达本节点的本地事务或外地事务立即引用到当前绳结中，从而形成本地到达序的证据，并在满足预设的本地到达绳同步触发条件时向其他共识节点同步所述本地到达绳中未同步部分，

接收、验证、存储、同步外地到达绳，解析外地到达绳获知外地到达序。其中，所述预设的本地到达绳同步触发条件，一般包括：

（1）当前结引入的是本地事务；

（2）可选的，未同步绳段的事务结或空结的个数达到了一定值；

（3）可选的，当前结的哈希值符合一定特征；

（4）可选的其他条件；

其中，从本地和外地到达绳中获知事务到达序的方法为：

$$K_{(T_0)}^{(n_0)} \prec K_{(T_1)}^{(n_0)} \in \Phi_A^{(n_0)} \Rightarrow T_0 \prec T_1 \in \mathrm{O}_A^{(n_0)}$$

优选地，为减轻验证外地到达绳的计算压力，到达绳处理子模块对外地到达绳进行随机、并行或协作验证；优选地，到达绳处理子模块为提供自适应的本地时间分辨率而并行构造本地到达绳；优选地，为减轻到达绳存储和通信压力，到达绳处理子模块分别对本地和外地到达绳数据进行压缩处理；优选地，到达绳处理子模块利用统计学方法分析本地和外地到达绳数据获得关于事务输入模块、共识网络和共识节点的特征变化轨迹，并利用所述特征作出相应的优化响应。

还原绳处理模块流程。共识节点之间接收和同步各节点记录的到达序信息，通过预设规则分析出每个到达序记录中蕴含的部分本真序信息，并将各部分本真序信息整合，从而各自还原出“还原序”。其中，共识节点的还原绳处理子模块根据预设规则从已获取的本地和外地到达绳中获取事务到达序信息，根据预设还原规则处理事务到达序信息输出两两事务的还原序和所有事务的还原序，同时构建本地还原绳存储还原序信息。优选地，为提高处理效率，共识节点的还原绳处理子模块并行地处理还原绳数据；优选地，为减轻存储压力，共识节点及时删除还原绳中的冗余时序数据。

优选地，共识节点优先判断事务共识时间相近的两两事务的还原序，再判断已写入还原绳的事务与未写入还原绳的事务之间的还原序。

共识绳处理模块流程。各共识节点之间接收和同步各节点的还原序信息，通过预设的理方法处理还原序信息，形成共识序。其中，共识节点的共识绳处理子模块多轮次交换还原序信息，对本地和外地还原绳进行共识处理，形成共识序。共识绳处理子模块构建本地共识绳存储共识序信息。优选地，为避免共识节点主观选择交换还原序信息的节点组合，共识绳处理子模块随机产生所述交换还原序信息的节点组合；优选地，共识绳处理子模块校验共识绳数据，保证共识绳不被篡改且尽量与其他共识节点的共识绳保持状态同步。

状态处理模块流程。共识节点按照共识序执行事务，使各状态数据副本产生一致的状态

变化轨迹。其中，共识节点的状态处理子模块存储、验证、同步状态数据，按照共识序解析本地共识绳中的事务，创建事务执行队列，将可执行的事务写入所述事务执行队列，调用事务对应的执行器执行所述事务执行队列中的事务，完成状态数据的状态变化，按照与状态获取模块协定的规则向状态获取模块输出状态数据。优选地，为提高事务执行效率，状态处理子模块并行执行事务；其中，状态处理子模块可以主动向状态获取模块推送订阅的状态数据，也可以响应状态获取模块的状态获取请求。

时间处理模块流程。共识节点利用共识序的状态变化轨迹定义、解释和推进一种时间。优选地，当事务定序系统 1 中产生了所述时间后，共识节点可以输入、输出和处理时间信息。其中，共识节点的时间处理子模块将本地共识绳的状态变化轨迹作为时间参考系来定义共识时间，所有共识节点按照一致的规则解释、推进共识时间，获取当前共识时间为网络中的事件盖共识时间戳，验证来自其他共识节点的共识时间戳，分析共识时间信息并作出优化响应处理；

状态获取模块流程。状态获取模块从共识网络中获取状态数据。其中，状态获取模块与共识节点协定状态获取规则，从共识节点获取状态数据并存储和处理所述状态数据。可选的，状态获取模块可以同时从多个共识节点获取状态数据并接受其中的多数结果，也可以接受其中的最新结果，也可以将多个结果合成为一个结果。

如图 33 所示，在一种实施方式中，一种共识节点解析外地到达绳获知外地到达序的方法。具体步骤为：外地到达绳解析子单元获取到待解析的外地到达绳段；外地到达绳解析子单元根据绳段的签名解析出所述绳段的源共识节点，在到达序信息存储单元中创建所述源共识节点和所述绳段的实体数据，创建所述源共识节点和所述绳段之间的关系数据，如果已经创建过上述数据则不重复创建；外地到达绳解析子单元获取所述外地到达绳段中的绳结，在到达序信息存储单元中为其中的每个事务结创建一个绳结实体数据，并创建所述绳结实体数据与所述外地到达绳实体数据之间的关系数据，不重复创建；外地到达绳解析子单元获取所述外地到达绳段中的事务引用，从事务存储模块中查找到对应的事务，如果找到了则为所述事务在到达序信息存储单元中创建一个事务实体数据，并创建所述事务实体数据与所述事务所在绳结实体数据之间的关系数据；如果没有找到，则当所述事务到达了本共识节点时，触发外地到达绳解析子单元为所述事务在到达序信息存储单元中创建上述实体数据和关系数据。不重复创建；

如图 34 和图 35 所示，在一种实施方式中，提供一种所述实体数据和关系数据的结构示意图。实体数据和关系数据一般用来表达到达绳的解析结果。所述实体数据包括：共识节点、到达绳、绳结、事务，实体数据一般存储所述实体的指针或 ID 以及必要的属性，所述关系数据包括：共识节点与到达绳的关系、到达绳与绳结的关系、绳结与事务的关系，关系数据可

以利用图、表、数组、二元组、JSON 对象等数据结构存储，便于查找和计算处理。其中，一个共识节点可能有多个到达绳，一个到达绳一般有多个绳结，一个绳结一般有一个或零个事务。在一种实施方式中，(:节点）-[:创建]->（:到达绳）-[:包括]->（:绳结）-[:引入]->（:事务)，以所述结构保存在 Neo4j 图数据库中，通过 CypherQL 语言查找和修改。

一种外地到达绳拼接方法包括：外地到达绳拼接子单元判断是否有待拼接的外地到达绳段，如果有则执行下一步，如果没有则等待下一次执行；外地到达绳拼接子单元获取到待拼接的外地到达绳段 L0，并获取到来自所述绳段的源共识节点的已拼接完成的外地到达绳段 L1；外地到达绳拼接子单元判断 L0 与 L1 是否有重叠部分并完成对应的拼接；只要有待拼接的外地到达绳段。

例如，L0 中包括 99-105 号绳结，L1 中包括 0-101 号绳结，则 99、100、101 号绳结组成的绳段为重叠部分。如果有重叠部分，判断重叠部分的对应绳结是否完全一样，例如，判断 L0 中的 99 号绳结和 L1 中的 99 号绳结数据是否完全一样，以此类推 100 和 101 号绳结。如果存在重叠部分且重叠部分存在不一致情况，则所述 L0 为错误绳结，应当舍弃；如果存在重叠部分且重叠部分完全一样，则舍弃 L0 中的重叠部分，将 L0 中未重叠部分拼接到 L1 的尾部，例如上述例子，如果 99、100、101 号绳结完全对应一样，则舍弃所述三个绳结，将 L0 中的第 102-105 号绳结拼接到 L1 的 101 号绳结之后；可选的，如果不存在重叠部分但是 L1 的末尾绳结通过连续的打空结可以生成到 L0 的首个绳结，则判断 L1 与 L0 可接续，例如：L0 包括 103-110 号绳结，L1 包括 0-100 号绳结，如果按照打空结的方法，L1 从 100 号打出 101、102、103 号绳结，发现 L1 的 103 号绳结和 L0 的 103 号绳结数据完全一致，则说明 L1 和 L0 可拼接，其中未重叠部分为空结，拼接后 L1 包括 0-110 号绳结；此外，如果不存在重叠部分，且已知未重叠部分存在事务结，则将 L0 暂存到外地到达绳获取模块仍作为待拼接的外地到达绳段。

在一种实施方式中，提供了一种共识节点为提供自适应的本地时间分辨率而并行构造本地到达绳的方法，到达绳中所有绳结组成单链结构，就好像是单股的结绳，称作单股结绳。单股结绳中，绳结只能同步地串行生成，限制了事务的写入速度和处理资源的利用率。为了充分利用多核处理资源以及提高到达绳的时间分辨率和对到达事务的处理效率，提出一种多核并行打结方法，包括：方式一：本地到达绳打结子单元在多核资源上实施 N 个待执行的打结器和一个调度器；所述调度器在每个轮次中依次异步调用一个空闲的打结器进行打结操作，每次调用前，调度器生成绳结编号、确定前序结，并确定是否有待引入绳结的事务，调度器将绳结编号、前序结、待引入事务输入给打结器，打结器按照调度器的输入进行打结。

方式二：本地到达绳打结子单元在多核资源上实施 N 个待执行的打结器，N 个打结器各自有唯一的编号并持续竞争一个互斥锁，所述互斥锁携带一个整数变量 a 用于生成绳结编号

和一个变量 b 用于记录最近一次竞争到互斥锁的打结器的编号；当其中一个打结器竞争到互斥锁后，将互斥锁所携带的整数变量 a 加 1 从而生成当前绳结编号，然后读取互斥锁变量 b 从而获取上一个竞争到互斥锁的打结器的编号，然后将变量 b 置为本打结器的编号，然后检查是否有待引入绳结的事务，如果有则锁定所述事务并决定打事务结，如果没有则决定打空结，然后立即释放互斥锁并开始打结，所打绳结的前序结为本打结器上一次打的绳结，所打绳结还要包括本打结器的编号、上一个竞争到互斥锁的打结器的编号。打结完成后继续竞争互斥锁。

可见，按照上述方法，N 个打结器将产生 N 股的结绳，可以称 N 为结绳的并行度，将并行度为 N 的结绳称为 N 股结绳。进一步地，为了增大结绳被重新构造的难度，可以设置两个前序结，其中一个前序结为本打结器的最近一次打出的绳结，另一个为上一次获得打结权利的打结器，在本打结器获取到打结权利之前最近一次打出的绳结。在上述方式一中，当打结器被调度器调用即可视作打结器获得了打结权利；在方式二中，打结器竞争到了互斥锁则视作获得了打结权利。进一步地，由于不同并行度的打结占用的计算资源不同、实现的时间分辨率也不同。到达频率不高时，过高的打结并行度是一种浪费；而当事务到达频率较高时，较低的打结并行度则不能满足时序的分辨率。为了根据事务到达频率而自适应地调整打结的并行度，提出一种自适应可变并行度的到达绳打结方法及到达绳验证方法，包括：一个监控器获取本地到达绳中包含有当前结的一个绳段，统计所述绳段中事务结的个数占所述绳段绳结总数的比例，当所述比例超过某个预设值时，增加本地到达绳的并行度，当所述比例低于某个预设值时，降低本地到达绳的并行度。可以称并行度有变化的结绳为变股结绳。当共识节点采用 N 股结绳或者变股结绳方式构造本地到达绳时，本地到达绳的同步触发机制、验证机制也需要进行适应的改变。在本申请中不做赘述。

在一种实施方式中，一种共识节点分别对本地到达绳数据进行压缩处理的方法。具体步骤为：到达绳处理优化单元从本地到达绳存储子单元中获取本地到达绳数据；到达绳处理优化单元保留本地到达绳的部分绳结，向本地到达绳存储子单元发出请求删除其他绳结。其中，方式一：保留初始绳结、事务结、末尾绳结、部分空结；方式二：保留初始绳结、事务结、未同步的全部绳段；方式三：保留初始绳结、事务结、末尾绳结、以及某个整数的整数倍编号的绳结。此外，还可以有其他方式，目的在于保留基本信息，删除可根据基本信息导出的数据。在一种实施方式中，一个本地到达绳中，包括以下绳结，（0：初始绳结）（1：事务结）（2：空结）（3：空结）（4：空结）（5：事务结）（6：末尾绳结，事务结），从 4 号开始未同步。按照第一种方式，一种结果是保留 0、1、2、5、6 号绳结；按照第二种方式，保留 0、1、4、5、6 号绳结；按照第三种方式，保留 3 的整数倍，则保留 0、1、3、5、6 号绳结。

在一种实施方式中，一种共识节点分别对外地到达绳数据进行压缩处理的方法。具体步

骤为：到达绳处理优化单元从外地到达绳存储子单元中获取本地到达绳数据；到达绳处理优化单元保留外地到达绳的部分绳结，向外地到达绳存储子单元发出请求删除其他绳结。其中，方式一：保留初始绳结、事务结、末尾绳结、部分空结；方式二：保留初始绳结、事务结、未同步的全部绳段；方式三：保留初始绳结、事务结、末尾绳结、以及某个整数的整数倍编号的绳结。此外，还可以有其他方式，目的在于保留基本信息，删除可根据基本信息导出的数据。在一种实施方式中，一个本地到达绳中，包括以下绳结，（0：初始绳结）（1：事务结）（2：空结）（3：空结）（4：空结）（5：事务结）（6：末尾绳结，事务结），从 4 号开始未同步。按照第一种方式，一种结果是保留 0、1、2、5、6 号绳结；按照第二种方式，保留 0、1、4、5、6 号绳结；按照第三种方式，保留 3 的整数倍，则保留 0、1、3、5、6 号绳结。

在一种实施方式中，一种共识节点利用统计学方法分析本地和外地到达绳数据获得关于事务输入模块、共识网络、共识节点的特征变化轨迹，并利用所述特征作出相应的优化响应的方法。其中，具体步骤为：到达序信息分析单元预先设置一个或一组分析项，每个分析项是一个主动或被动执行的分析程序；到达序信息分析单元，从到达序信息存储单元获取到达序信息并执行预设的分析步骤，输出分析结果给到达绳处理优化单元；到达绳处理优化单元获取到一项到达绳解析数据的分析结果，按照预设的关联关系获取到所述分析项对应的一个或一组优化响应程序，并执行所述优化响应程序；

在一种实施方式中，预先设置一个分析项：通过本地和外地到达绳了解事务到达共识节点频率，则，步骤包括：

（1）选取一段到达绳，所述到达绳可以是本地到达绳，也可以是外地到达绳；

（2）获得该段到达绳所包含的绳结数量 a，具体可以是用 S1 所述该段到达绳的末尾绳结编号减去起始绳结编号；

（3）统计 S1 所述该段到达绳中的事务结的个数 b0，相对于创建该到达绳的共识节点的本地事务结个数 b1，以及相对于创建到达绳的共识节点的外地事务 b2；

（4）则相对于创建该到达绳的共识节点来说，在 S1 所述该段到达绳所代表的历史时间内，事务到达共识节点的频率为 b0/a，本地事务到达共识节点的频率为 b1/a，外地事务到达共识节点的频率为 b2/a；

在一种实施方式中，当本地到达绳是可变并行度的打结延长方式时，到达序信息分析单元会统计一段时间内事务到达频率的变化情况并形成分析结果，到达绳处理优化单元获取到所述分析结果，按照预设关联，所述分析项对应着到达绳并行度变化响应规则，于是执行所述响应规则，所述响应规则的执行过程为，获取所述分析结果，发现事务到达频率超过了某个阈值，则响应结果为增大所述本地到达绳的并行度。

如图 36 所示，在一种实施方式中，一种共识节点根据预设规则从已获取的本地和外地到达绳中获取事务到达序信息的方法。具体步骤为：到达序信息获取单元从到达序信息存储单元中获取外地到达序信息；到达序信息获取单元从本地到达绳存储子单元中获取本地到达序信息，所述本地到达序信息与外地到达序信息的结构相同，不同的是，所述本地到达序信息中的节点为本节点，事务为到达本节点的事务；其中，从本地和外地到达绳中获知事务到达序信息的原理表达为，在同一个到达绳绳中，事务所在绳结编号的大小顺序与事务到达所述到达绳的本地节点的到达序相同。记作：

$$K_{(T_0)}^{(n_0)} \prec K_{(T_1)}^{(n_0)} \in \Phi_A^{(n_0)} \Rightarrow T_0 \prec T_1 \in \mathrm{O}_A^{(n_0)}$$

如图 37 所示，在一种实施方式中，一种共识节点根据预设还原规则处理事务到达序信息输出两两事务的还原序和所有事务的还原序的方法。具体步骤为：本地还原绳生成单元从到达序信息获取单元中获取来自到达序信息；本地还原绳生成单元处理所述到达序信息，输出两两事务的还原序和所有事务的还原序。其中，所述预设还原规则，至少包括：

（1）规则 1：待共识事务的还原序晚于已共识事务；

$$T_0 \in \mathrm{O}_C \wedge T_1 \notin \mathrm{O}_C \Rightarrow \left(T_0 \prec T_1\right) \in \mathrm{O}_R$$

在规则 1 中，已共识的时序不可逆转，因此置信度为 100%；

（2）规则 2：同一个节点的两个本地事务的到达序与还原序相同；

$$\left(T_0^{(n_i)} \prec T_1^{(n_i)}\right) \in \mathrm{O}_A^{(n_i)} \Rightarrow \left(T_0 \prec T_1\right) \in \mathrm{O}_R$$

在规则 2 中，两个事务的到达时间间隔越大，置信度越高，置信度的值可以采用数理统计相关知识计算获得，本申请不再赘述；

（3）规则 3：已知事务 T0 为节点 j 的本地事务，事务 T1 为节点 i 的本地事务，如果 T0 先比 T1 到达节点 i，且 T0 先比 T1 到达节点 j，则有 T0 还原序早于 T1，记作：

$$\left(T_0^{(n_j)} \prec T_1^{(n_i)}\right) \in \mathrm{O}_A^{(n_i)} \wedge \left(T_0^{(n_j)} \prec T_1^{(n_i)}\right) \in \mathrm{O}_A^{(n_j)} \Rightarrow \left(T_0 \prec T_1\right) \in \mathrm{O}_R$$

（4）在规则 3 中，所比较的两个事务如果在同一个到达绳中所在绳结编号的差越大，则根据规则 1 判断的结果的置信度越大，置信度的值可以采用数理统计相关知识计算获得，本申请不再赘述；

（5）规则 3-1：对于规则 3 有一个弱规则，即只考察本地到达序就作出判断，表达为：

$$\left(T_0^{(n_j)} \prec T_1^{(n_i)}\right) \in \mathrm{O}_A^{(n_i)} \Rightarrow \left(T_0 \prec T_1\right) \in \mathrm{O}_R$$

（6）规则 4：对于事务 T0 和 T1，在同一个节点的本地到达绳中，如果 T0 先于 T1 被引入绳结，则说明 T0 先于 T1 到达所述节点，统计所有的已获取的到达绳中 T0 和 T1 的到达先后关系，如果 T0 的先到达节点数减去 T1 的先到达节点数所得的差值再除以节点总数所得的比值大于某个阈值，则 T0 还原序早于 T1，记作：

$$\frac{1}{|V|}\left(\sum_{i}^{|V|}\left(T_0 \prec T_1\right)_A^{(n_i)}-\sum_{i}^{|V|}\left(T_1 \prec T_0\right)_A^{(n_i)}\right)\geq \varepsilon,\ \ \varepsilon\in(0,1]\Rightarrow\left(T_0 \prec T_1\right)\in \mathrm{O}_R$$

在规则 4 中，所比较的两个事务如果先到达节点数的差距越大，则根据规则 2 判断的结果的置信度越大；

（7）规则 5：事务的还原序满足传递性。

$$\left(T_0 \prec T_1\right)\in \mathrm{O}_R \wedge \left(T_1 \prec T_2\right)\in \mathrm{O}_R \Rightarrow \left(T_0 \prec T_2\right)\in \mathrm{O}_R$$

在规则 5 中，事务之间可传递的事务越多，则置信度越高，置信度的值可以采用数理统计相关知识计算获得，本申请不再赘述；

（8）规则 6：同一个节点的两个到达事务（无论本地外地）的到达时间间隔大于某个阈值。其中，阈值大小一般大于共识延迟；

$$\left(T_0 \prec T_1\right)\in \mathrm{O}_A^{(n_i)} \wedge \left(t\left(T_1\right)_A^{(n_i)}-t\left(T_0\right)_A^{(n_i)}>\lambda\tau\right)\Rightarrow\left(T_0 \prec T_1\right)\in \mathrm{O}_R$$

（9）在规则 6 中，两个事务的到达时间间隔越大，置信度越高，置信度的值可以采用数理统计相关知识计算获得，本申请不再赘述；

（10）规则 6-1：对于所述规则 6，在尚不能估计共识延迟的时候，可以计算所述两个事务在同一个到达绳的绳结编号的差值与间隔最大的两个事务的绳结编号的差值的比值，相加所有到达绳的计算结果，如果为负值，则作为被减数位置的事务的还原序更早，即：

$$\sum_{i}^{|V|}\frac{\left\|K_{(T_0)}-K_{(T_1)}\right\|_A^{(n_i)}}{\max\left\|K_{(T_j)}-K_{(T_k)}\right\|_A^{(n_i)}}<0\Rightarrow\left(T_0 \prec T_1\right)\in \mathrm{O}_R$$

（11）规则 6-2：对于所述规则 6-1，也可以不计算间隔最大的两个事务的绳结编号的差值，而是选择间隔较大的两个事务的绳结编号的差值。

（12）规则 6-3：对于所述规则 6-1，两个节点可以自行相互多次通信提供本地到达绳的当前结编号，在接收到对方本地到达绳当前结编号后立即回复本节点的本地到达绳当前结编号，计算连续两次回复的结编号的差值与连续两次接收的结编号的差值的比值，得到两个节点的本地到达绳打结频率的比值。选定一个节点的本地到达绳为基准达到绳，将待定序的两个事务的到达绳结编号的差值换算为基准达到绳的差值，再求和判断还原序。

（13）规则 7：可选的，利用事务同步的路径信息而设置的判断规则，例如：

已知 T0 从入口节点出发途径 N 个节点到达本地，而 T1 从入口节点出发途径 M 个节点到达本地，如果在本地到达序中，有 T1 早于 T0 到达，且 M>N，则判断 T1 更大概率还原序早于 T0，其中，M-N 的值越大，置信度越高；

（14）规则 8：上述条件的组合条件。其中，对于同一对事务，当利用上述规则获得的结果发生冲突时（例如通过规则 1 得出 T0 早于 T1，而通过规则 2 得出 T1 早于 T0，则冲突），选择多数判断结果（例如通过规则 1、2、3、4 得出 T0 早于 T1，而通过规则 5、6 得出 T0 早于 T0，则多数结果为 T0 早于 T1，最终选择多数结果判断 T0 早于 T1）。优选地，对于冲突结果，判断相关节点的到达绳数据是否伪造时序并做相应处理。

例如，一个事务定序共识系统，包括一个共识网络 G，若干个事务输入模块 I 和若干个状态获取模块 F。其中，

事务输入模块有 3 个，记作：

$$I=\{i_0,i_1,i_2\}$$

状态获取模块有 3 个，记作：

$$F=\{f_0,f_1,f_2\}$$

节点 0、1、2、3 组成一个共识网络，如图 4 所示，记作：

$$G=(V,E)$$
$$V=\{n_0,n_1,n_2,n_3\}$$
$$E=\{\langle n_0,n_1\rangle,\langle n_0,n_2\rangle,\langle n_1,n_2\rangle,\langle n_1,n_3\rangle,\langle n_2,n_3\rangle\}$$

为了便于计算和说明，假设各个节点之间的延迟 D 分别为一个固定值，具体为：

$$D(n_0,n_1)=3,D(n_0,n_2)=2,D(n_1,n_2)=2,D(n_1,n_3)=4,D(n_2,n_3)=2$$

为了演示一组事务在共识网络中被共识的过程，假设客观存在一个本真序，

$$O_T=T_0\prec T_1\prec T_2\prec T_3\prec T_4\prec T_5$$

并且，假设存在一种绝对时间，按照绝对时间和本真序设计一个事务进入网络的计划表。其中，本真序中的事务由事务输入模块在指定时刻输入给指定节点。例如，在时刻 10，事务输入模块 1 以节点 0 为入口节点输入事务 T0，以此类推如下：

$$t(i_1:T_0\rightarrow n_0)=10.0$$
$$t(i_0:T_1\rightarrow n_1)=12.8$$
$$t(i_1:T_2\rightarrow n_2)=14.6$$
$$t(i_0:T_3\rightarrow n_3)=16.4$$
$$t(i_1:T_4\rightarrow n_0)=18.2$$
$$t(i_0:T_5\rightarrow n_1)=20.0$$

开始计时后，按照计划表由指定事务输入模块将指定事务输入共识网络的指定入口节点。假设 t=0 时，各节点的本地到达绳开始打结，分别如下：

$$\Phi_A^{(n_0)} = K_{a(n_0)}^{(0)}$$
$$\Phi_A^{(n_1)} = K_{a(n_1)}^{(0)}$$
$$\Phi_A^{(n_2)} = K_{a(n_2)}^{(0)}$$
$$\Phi_A^{(n_3)} = K_{a(n_3)}^{(0)}$$

由于各节点打结频率不同，当 t=10 时，各节点的本地到达绳如下，

$$\Phi_A^{(n_0)} = K_{a(n_0)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(1000)}$$
$$\Phi_A^{(n_1)} = K_{a(n_1)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(2000)}$$
$$\Phi_A^{(n_2)} = K_{a(n_2)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(3000)}$$
$$\Phi_A^{(n_3)} = K_{a(n_3)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(4000)}$$

此时，T0 被入口节点 0 获取，节点 0 判断 T0 为本地事务，验证、解析、存储和同步本地事务 T0。节点 0 将事务 T0 打结到本地到达绳的 1001 号绳结，记作：

$$\Phi_A^{(n_0)} = K_{a(n_0)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(1000)} \prec K_{(T_0)}^{(1001)}$$

此时，节点 0 将本地到达绳的未同步绳段进行同步。

当 t=11 时，由节点 0 同步的 T0 到达了节点 1，节点 1 判断 T0 为外地事务，验证、解析、存储和同步外地事务 T0。此时，节点 1 的本地到达绳末尾编号为 2200，于是节点 1 将事务 T0 打结到本地到达绳的 2201 号绳结，记作：

$$\Phi_A^{(n_1)} = K_{a(n_1)}^{(0)} \prec K_{\varnothing}^{(1)} \prec \cdots \prec K_{\varnothing}^{(2200)} \prec K_{(T_0)}^{(2201)}$$

此时，节点 1 将本地到达绳的未同步绳段进行同步。

经过一段时间后，各事务都已经输入到了共识网络中且同步到各节点，且各节点的本地到达绳也都同步到了共识网络的其他节点。

为简化计算，将节点为了同步一个事务而在本地产生的处理时间忽略不计，则根据节点之间的延迟设定可以知道事务到达各个节点的“绝对时间”为：

$$t(T_0 \to n_0) = 10,\quad t(T_0 \to n_1) = 13,\quad t(T_0 \to n_2) = 12,\quad t(T_0 \to n_3) = 14$$
$$t(T_1 \to n_0) = 15.8,\quad t(T_1 \to n_1) = 12.8,\quad t(T_1 \to n_2) = 14.8,\quad t(T_1 \to n_3) = 16.8$$
$$t(T_2 \to n_0) = 16.6,\quad t(T_2 \to n_1) = 16.6,\quad t(T_2 \to n_2) = 14.6,\quad t(T_2 \to n_3) = 16.6$$
$$t(T_3 \to n_0) = 20.4,\quad t(T_3 \to n_1) = 20.4,\quad t(T_3 \to n_2) = 18.4,\quad t(T_3 \to n_3) = 16.4$$
$$t(T_4 \to n_0) = 18.2,\quad t(T_4 \to n_1) = 21.2,\quad t(T_4 \to n_2) = 20.2,\quad t(T_4 \to n_3) = 22.2$$
$$t(T_5 \to n_0) = 23,\quad t(T_5 \to n_1) = 20,\quad t(T_5 \to n_2) = 22,\quad t(T_5 \to n_3) = 24$$

各节点的到达绳数据如下：

$$\Phi_A^{(n_0)} = K_{a(n_0)}^{(0)} \prec K_{(T_0)}^{(1001)} \prec K_{(T_1)}^{(1581)} \prec K_{(T_2)}^{(1661)} \prec K_{(T_4)}^{(1821)} \prec K_{(T_3)}^{(2041)} \prec K_{(T_5)}^{(2301)}$$
$$\Phi_A^{(n_1)} = K_{a(n_1)}^{(0)} \prec K_{(T_1)}^{(2561)} \prec K_{(T_0)}^{(2601)} \prec K_{(T_2)}^{(3321)} \prec K_{(T_5)}^{(4001)} \prec K_{(T_3)}^{(4081)} \prec K_{(T_4)}^{(4241)}$$
$$\Phi_A^{(n_2)} = K_{a(n_2)}^{(0)} \prec K_{(T_0)}^{(3601)} \prec K_{(T_2)}^{(4381)} \prec K_{(T_1)}^{(4441)} \prec K_{(T_3)}^{(5521)} \prec K_{(T_4)}^{(6061)} \prec K_{(T_5)}^{(6601)}$$
$$\Phi_A^{(n_3)} = K_{a(n_3)}^{(0)} \prec K_{(T_0)}^{(5601)} \prec K_{(T_3)}^{(6561)} \prec K_{(T_2)}^{(6641)} \prec K_{(T_1)}^{(6721)} \prec K_{(T_4)}^{(8881)} \prec K_{(T_5)}^{(9601)}$$

每个节点在接收到外地到达绳后，就开始进行还原绳处理。以节点 0 为例，根据本地和外地到达绳可以获知事务到达各节点的时序，如下：

$$\mathrm{O}_A^{(n_0)} = T_0 \prec T_1 \prec T_2 \prec T_4 \prec T_3 \prec T_5$$
$$\mathrm{O}_A^{(n_1)} = T_1 \prec T_0 \prec T_2 \prec T_5 \prec T_3 \prec T_4$$
$$\mathrm{O}_A^{(n_2)} = T_0 \prec T_2 \prec T_1 \prec T_3 \prec T_4 \prec T_5$$
$$\mathrm{O}_A^{(n_3)} = T_0 \prec T_3 \prec T_2 \prec T_1 \prec T_4 \prec T_5$$

根据还原规则 1，目前尚未有已共识事务，无法利用所述规则 1。

根据还原规则 2，已知，节点 0 的本地事务有 T0 和 T4，节点 1 的本地事务有 T1 和 T5，节点 2 的本地事务有 T2，节点 3 的本地事务有 T3，则，

$$\left(T_0^{(n_0)} \prec T_4^{(n_0)}\right) \in \mathrm{O}_A^{(n_0)} \Rightarrow \left(T_0 \prec T_4\right) \in \mathrm{O}_R$$

$$\left(T_1^{(n_1)} \prec T_5^{(n_1)}\right) \in \mathrm{O}_A^{(n_1)} \Rightarrow \left(T_1 \prec T_5\right) \in \mathrm{O}_R$$

根据还原规则 3，

$$\left(T_1^{(n_1)} \prec T_4^{(n_0)}\right) \in \mathrm{O}_A^{(n_0)} \wedge \left(T_1^{(n_1)} \prec T_4^{(n_0)}\right) \in \mathrm{O}_A^{(n_1)} \Rightarrow \left(T_1 \prec T_4\right) \in \mathrm{O}_R$$
$$\left(T_2^{(n_2)} \prec T_4^{(n_0)}\right) \in \mathrm{O}_A^{(n_0)} \wedge \left(T_2^{(n_2)} \prec T_4^{(n_0)}\right) \in \mathrm{O}_A^{(n_2)} \Rightarrow \left(T_2 \prec T_4\right) \in \mathrm{O}_R$$
$$\left(T_0^{(n_0)} \prec T_5^{(n_1)}\right) \in \mathrm{O}_A^{(n_1)} \wedge \left(T_0^{(n_0)} \prec T_5^{(n_1)}\right) \in \mathrm{O}_A^{(n_0)} \Rightarrow \left(T_0 \prec T_5\right) \in \mathrm{O}_R$$
$$\left(T_2^{(n_2)} \prec T_5^{(n_1)}\right) \in \mathrm{O}_A^{(n_1)} \wedge \left(T_2^{(n_2)} \prec T_5^{(n_1)}\right) \in \mathrm{O}_A^{(n_2)} \Rightarrow \left(T_2 \prec T_5\right) \in \mathrm{O}_R$$
$$\left(T_0^{(n_0)} \prec T_2^{(n_2)}\right) \in \mathrm{O}_A^{(n_2)} \wedge \left(T_0^{(n_0)} \prec T_2^{(n_2)}\right) \in \mathrm{O}_A^{(n_0)} \Rightarrow \left(T_0 \prec T_2\right) \in \mathrm{O}_R$$
$$\left(T_0^{(n_0)} \prec T_3^{(n_3)}\right) \in \mathrm{O}_A^{(n_3)} \wedge \left(T_0^{(n_0)} \prec T_3^{(n_3)}\right) \in \mathrm{O}_A^{(n_3)} \Rightarrow \left(T_0 \prec T_3\right) \in \mathrm{O}_R$$

根据还原规则 5，结合规则 2 和 3 的结果，

$$\left(T_0 \prec T_2\right) \in \mathrm{O}_R \wedge \left(T_2 \prec T_5\right) \in \mathrm{O}_R \Rightarrow \left(T_0 \prec T_2 \prec T_5\right) \in \mathrm{O}_R$$
$$\left(T_0 \prec T_2\right) \in \mathrm{O}_R \wedge \left(T_2 \prec T_4\right) \in \mathrm{O}_R \Rightarrow \left(T_0 \prec T_2 \prec T_4\right) \in \mathrm{O}_R$$

目前还不确定 T0 与 T1，T1 与 T2，T1 与 T3，T2 与 T3，T2 与 T4，T3 与 T4，T4 与 T5 的还原序关系。

根据还原规则 4：对于 T0 与 T1 来说，在节点 0、2、3 中均有 T0 早于 T1 到达，只有节点 1 是 T1 早于 T0，则判断 T0 早于 T1 进入网络，同理可以判断出 T1 早于 T3，T2 早于 T3，T2 早于 T4，T3 早于 T4，T4 早于 T5，此时再根据规则 5，有，

$$(T_0 \prec T_1 \prec T_3 \prec T_4 \prec T_5) \in \mathrm{O}_R$$

$$(T_0 \prec T_2 \prec T_3 \prec T_4 \prec T_5) \in \mathrm{O}_R$$

但是，T1 和 T2 各有两个节点的到达序早于对方，目前还不确定 T1 与 T2 的还原序关系。

根据还原规则 6，获取 T0 和 T1 在本地和外地到达绳的绳结编号并计算。

$$\Phi_A^{(n_0)} : \frac{\left\| K_{(T_1)} - K_{(T_2)} \right\|}{\left\| K_{(T_0)} - K_{(T_5)} \right\|} = \frac{-80}{1300}, \quad \Phi_A^{(n_1)} : \frac{\left\| K_{(T_1)} - K_{(T_2)} \right\|}{\left\| K_{(T_1)} - K_{(T_4)} \right\|} = \frac{-760}{1680},$$

$$\Phi_A^{(n_2)} : \frac{\left\| K_{(T_1)} - K_{(T_2)} \right\|}{\left\| K_{(T_0)} - K_{(T_5)} \right\|} = \frac{60}{3000}, \quad \Phi_A^{(n_3)} : \frac{\left\| K_{(T_1)} - K_{(T_2)} \right\|}{\left\| K_{(T_0)} - K_{(T_5)} \right\|} = \frac{80}{4000},$$

$$\frac{-80}{1300} + \frac{-760}{1680} + \frac{60}{3000} + \frac{80}{4000} \approx -4.526 < 0 \Rightarrow (T_1 \prec T_2) \in \mathrm{O}_R$$

至此，节点 0 可以得出还原序，

$$(T_0 \prec T_1 \prec T_2 \prec T_3 \prec T_4 \prec T_5) \in \mathrm{O}_R$$

同理，节点 1、2、3 根据相同的方法可以得出还原序，

$$(T_0 \prec T_1 \prec T_2 \prec T_3 \prec T_4 \prec T_5) \in \mathrm{O}_R$$

节点 0 将本地还原序发送给其他节点，其他节点回复本节点的本地还原序，节点 0 对比所收到的外地还原序，发现完全一致，则认定还原序达成共识序。

节点 0 将共识序按顺序写入共识绳，则得到共识绳为，

$$\Phi_C^{(n_0)} = K_{init}^{(0)} \prec K_{(T_0)}^{(1)} \prec K_{(T_1)}^{(2)} \prec K_{(T_2)}^{(3)} \prec K_{(T_3)}^{(4)} \prec K_{(T_4)}^{(5)} \prec K_{(T_5)}^{(6)}$$

节点 0 按共识序解析共识绳中的事务，写入事务执行队列，并逐一调取事务对应的事务执行器执行事务执行队列中的事务，完成对状态数据的变更。其他节点也是做相同的操作。例如，所述事务的待执行内容分别为：

T0:要求对变量 A 加 1；

T1:要求对变量 A 加 2；

T2:要求对变量 B 从 2 加 3；

T3:要求对变量 A 减 3；

T4:要求对变量 A 减 2；

T5:要求对变量 B 从 2 加 1；

存在约束条件：A 和 B 不能小于 0；

在所述事务输入网络之前，节点的状态数据包括：

A：值为 1；

B：值为 2。

当共识序开始执行后：

执行 T0，A 的值变为 1+1=2；

执行 T1，A 的值变为 2+2=4；

执行 T2，B 的值为 2，从 2 加 3 变为 5；

执行 T3，A 的值变为 4-3=1；

执行 T4，A 的值 1-2<0，不符合所述约束条件，T4 舍弃；

执行 T5，B 的值为 5，无法从 2 加 1，T5 舍弃；

同时，状态获取模块从任意节点中获取状态数据。例如：

状态获取模块 0 向节点 0 请求 A 的值，A 向节点 0 返回此时 A 的值；

状态获取模块 1 已向节点 2 订阅了 B 的值，每当 B 的值更新时，节点 2 就向所述状态获取模块 1 推送 B 的值；

状态获取模块 2 同时从节点 0、1、2、3 请求 A 的值；

状态获取模块 3 同时从节点 1、2 订阅 A 的值；

在一种实施方式中，构建本地还原绳存储还原序信息的方法。构建一个拓扑图数据结构，所述拓扑图数据结构用于存储和表示本共识节点从所接收的到达绳中分析还原出待共识事务进入网络的时序关系，该图称作“还原绳”。其中，所述还原绳中的“点”代表实体，“边”代表实体之间的关系，点和边均有类型和属性。点的类型包括：事务、共识节点；边的类型包括：发布关系（1 条边，连接 1 个事务实体和发布该事务的 1 个共识节点实体）、时序关系（1 条边，连接两个事务 A 和 B，属性为共识节点判断 A 事务早于 B 事务进入网络的确信分值，所述确信分值计算得到）；共识节点从待共识事务集合中选择两个事务 A 和 B，按照分析规则得出 A 早于 B 进入网络的确信分值，将分析结果写入还原绳，同时删除冗余的时序关系边；共识节点将还原绳中认为已稳定且未完成共识的部分与其他共识节点进行共识处理，并将已完成共识的部分按照事务进入网络的共识时序写入共识绳；将已完成共识的事务在从还原绳中标记为“已共识”。可选的，为了节省存储空间，可以将已完成共识的事务、发布关系和时序关系从还原图中删除。

如图 38 所示，在一种实施方式中，一种共识节点多轮次交换还原序信息，对本地和外地

还原绳进行共识处理，形成共识序的方法，具体步骤为：共识节点在还原绳中截取未共识的且已稳定的还原绳段以及当前的共识绳末尾绳结，随机向 N 个共识节点发送。所述共识节点在还原绳中截取未共识的且已稳定的还原绳段的方法，包括：确认所有的接收的到达绳中包含的事务均已分析完毕；删除掉冗余关系后，还原绳中较早进入网络的事务的时序关系形成一条单链确认不可能有新的事务插入到所述单链中的较早部分，则此部分单链为稳定时序；共识节点随机向 N 个共识节点请求共识已稳定的还原绳段及共识绳末尾绳结；共识节点接收到其他节点发来的待共识绳段后，与本共识节点的对应绳段进行比对，对于其中不相同的时序关系，接受其中的多数结果，例如：超过三分之二，并将更新后的待共识绳段继续发出；重复直到所接受的所有结果都有一致的时序关系，则认为已就所述结果形成共识。

在一种实施方式中，提供了一种共识节点随机产生所述交换还原序信息的节点组合的方法所述方法的具体步骤为：获取节点信息，根据到达绳、还原绳和共识绳的数据获取一个初始值；根据所述初始值生成随机数；根据节点信息和所述随机数产生节点组合。

在一种实施方式中，一种共识节点及时删除还原绳中的冗余时序数据的方法。具体步骤为：判断出并删除待共识事务中的冗余时序关系边。例如，在还原绳中存在待共识事务 A、B、C、D，时序关系边 A->B，B->C，C->D，A->D，则 A->D 为冗余的时序关系边；判断出并删除已共识事务和待共识事务之间的时序关系边。例如，在还原绳中存在待共识事务 A 和已共识事务 B，则 B->A 为冗余的时序关系边；

在一种实施方式中，一种共识节点优先判断事务共识时间相近的两两事务的还原序，再判断已写入还原绳的事务与未写入还原绳的事务之间的还原序的方法。具体步骤为：将待还原的事务按照时间戳的顺序排序；按照计算机知识的排序算法流程对所述事务的还原序排序；例如，所述计算机知识的排序算法包括：冒泡排序、快速排序、分治排序等。

在一种实施方式中，一种共识节点构建本地共识绳存储共识序信息的方法。具体步骤为：共识节点加入共识网络后，触发共识绳初始化模块开始共识绳初始化同步；所述共识绳初始化同步模块向其他多个共识节点发出共识绳同步请求，所述请求的目的是获取到当前最新的共识绳结，其他共识节点回复最新的共识绳结；可选的，所述接受请求的节点，可以与发出请求的节点建立一段时间的长连接，持续将最新的共识绳结推送给发出请求的节点；可选的，共识绳初始化同步模块向其他多个共识节点发出请求下载指定的共识绳段，其目的是补足已在共识网络生成但是未在本地存储的共识绳。共识绳打结模块获取已共识的稳定还原绳段及所述绳段对应的共识绳结。将已经打入共识绳中的事务对应在本共识节点中标记为“已共识”，可以是每打入一个事务就标记一个事务，也可以是批量标记。

上述构建本地共识绳存储共识序信息的方法，在一种实施方式中，节点 0、1、2、3 构成一个共识网络。节点 4 加入共识网络后，向节点 0、1、2、3 发出共识绳同步请求，节点 0、

1、2、3 接收到请求后向节点 4 回复最新的共识绳结，其中，节点 0、1、2 均回复：当前最新绳结为编号 199，引入事务为 T1；节点 3 回复：当前最新绳结为 198，引入事务为 T0。于是节点 4 接受多数一致的结果，即最新绳结为编号 199，引入事务为 T1，并且验证了节点 3 回复的 198 号绳结的数字摘要与节点 0、1、2 回复的 199 号绳结的前序结引用值相同，于是确信节点 3 并未伪造绳结。继而节点 4 分别向节点 0、1、2、3 发出下载指定共识绳段的请求，其中，向节点 0 请求 0-60 号共识绳结，向节点 1 请求 61-120 号共识绳结，向节点 2 请求 121-160 号共识绳结，向节点 4 请求 161-198 号共识绳结，然后节点 0、1、2、3 按请求回复指定绳段，节点 4 对下载的共识绳段分别做验证，验证通过后，将收到的绳段拼接为完整的共识绳段。当共识网络中的负载较大时，在节点同步最新共识绳结的过程中，可能已经产生了新的共识绳结，这种情况下，节点 4 可以与节点 0、1、2、3 建立长连接，节点 0、1、2、3 主动将新产生的共识绳结推送给节点 4。

其中，如果所述共识绳结未在本地共识绳中，且所述共识绳结的编号大于本地共识绳的末尾绳结的编号，则触发共识绳初始化同步模块从其他共识节点下载本地共识绳所缺失的绳段；如果所述共识结绳为本地共识绳的末尾绳结，则将所述已共识的稳定还原绳段中的事务按照由早到晚的顺序依次打结到共识绳中；如果所述共识绳结在本地共识绳中且不是末尾绳结，则检查所述已共识的稳定还原绳段的事务序列与本地共识绳中所述共识绳结及以后的事务序列是否从第一个绳结开始是连续的对应相同，如果是，则将已共识的稳定还原绳段的事务序列中去除所述连续的对应相同的绳段，将剩余绳段的事务按照由早到晚的顺序依次打结到共识绳中，而如果不是连续对应，则执行相关处理流程，一般的，这种情况不应当出现，如果出现则很有可能是遭到了攻击或者是本共识节点存在伪造共识绳的情况。

在一种实施方式中，一种共识节点并行执行事务的方法。具体步骤为：将共识绳中的未执行事务按照从早到晚的顺序，把相互之间有排队需求的事务分配在同一个执行队列中，而共识节点按照执行队列先进先出的顺序执行队列内的事务。共识节点可能同时存在多个执行队列，执行队列可以并行执行。在一种实施方式中，共识应用 A、B、C 和共识网络组成一个共识系统，在已知共识应用 A、B、C 的事务相互独立时，共识节点设置 3 个执行队列，其中，1 号队列对应共识应用 A 的事务，2 号对应 B，3 号对应 C，共识绳事务被按照由早到晚的顺序被分配到所述 3 个队列中，3 个队列并行执行，每个队列都按照先进先出的原则串行执行队列内的事务。

在一种实施方式中，共识节点定义、解释、推进共识时间的方法。具体步骤为：一个计数器，包括一个初始值，从共识绳的预设编号事务开始计时；共识绳中每新增一个事务，则计数器加 1 或者加一个正数；把计数器的当前读数作为当前网络内生时间的表示，称为共识时间。所述共识时间是整个共识网络的公共时间。

共识节点定义、解释、推进本地时间的方法。具体步骤为：共识节点把本地到达绳当做共识节点的内部时钟，将本地到达绳的绳结个数的增加作为时间推进的依据，将本地到达绳的绳结作为共识节点内部时间刻度。例如，在一个单股本地到达绳中，事务 T1 被引入 12 号绳结，事务 T2 被引入 19 号绳结，当前绳结的编号为 25，由于单股本地到达绳是串行延长的，每延长一个绳结就意味着一段时间流逝，如果把增加一个绳结记为推进了一个单位时间，则对于本共识节点来说，可以认为，T1 在 12 时刻到达，T2 在 19 时刻到达，T1 比 T2 早 19-12=7 个时间单位到达本地，当前时刻为 25，下一个时刻是 26。所述时间可以称作所述共识节点的本地时间。

本申请所述事务定序共识方法，本质上就是各个共识节点通过本地时间记录一组事务到达本共识节点的本地时间，继而就事务到达共识网络的时序达成共识，并利用事务到达共识网络的共识序定义共识网络的共识时间，再将共识时间和本地时间一起强化定序能力。

另外，由于不同共识节点的打结频率可能不同，所形成的本地时间的推进速度可能也不同，例如，当 1 号节点打了 10 个绳结，1 号节点的本地时间推进了 10 个单位，而 2 号节点在同一个时间段内打了 8 个绳结，1 号节点的本地时间推进了 8 个单位。

在一种实施方式中，一种共识节点获取当前共识时间的方法。具体步骤为：将共识绳的状态变化轨迹作为时间参考系定义共识时间；获取当前共识绳的状态，按照共识时间的解释方法来解释当前共识时间；按照共识时间的推进方法，更新当前共识时间；

在一种实施方式中，一种共识节点为网络中的事件盖共识时间戳的方法。具体步骤为：在有了共识时间后，网络内的事件就可以用共识时间作为时间戳。由于共识绳时间数字与共识绳的结绳 hash 值一一对应，可以用共识绳的结绳 hash 值作为共识绳时间的校验码，从而可以验证共识节点所提供共识时间戳的真伪。

时间戳用于标记某种行为发生的时间，本申请中所述时间戳并不是人类通常所使用的年月日时分秒时间表示，也不是由共识系统之外的时间源所提供的时间表示，而是以共识系统内部的某种递增数量为基础所形成的时间表示。例如：当本共识节点发生行为 A 时，时间戳获取模块获取到已共识事务的数量，假设数量为 101，则共识时间戳为 101，并为行为 A 打上共识时间戳 101；当共识节点获取到一个已打了共识时间戳（假设为 309）的行为 B 数据时，时间戳验证模块获取共识时间为 101，则由于 309 远大于 101，时间戳验证模块判定行为 B 的时间戳为假，并触发相关处理；当待共识事务 C 的时间戳为 20，而时间戳验证模块获取已共识事务的数量为 101，由于 20 远小于 101 且 C 为待共识事务，则时间戳验证模块判定事务 C 的时间戳为假，并触发相关处理。

在一种实施方式中，一种共识节点验证来自其他共识节点的共识时间戳的方法。具体步

骤为：获取所述被提供的时间戳，包括共识时间和校验码；根据所述校验码从本共识节点的共识绳中查找对应的绳结，获取所述绳结编号和所述绳结的 hash 值，计算出对应的共识时间和校验码，检查是否与被提供的时间戳相等，如果不相等则验证不通过，如果相等则验证通过；

在一种实施方式中，一种激励共识节点诚实进行共识处理的方法，提出一种激励共识节点诚实进行共识处理的方法，包括：入口共识节点为接收的本地事务 T 加盖共识绳时间 t0；当事务 T 被写入到共识绳时，获取当前的共识绳时间 t1，则共识延迟 Δt=t1-t0；获取所述事务 T 临近的一组已共识事务的共识延迟的区间[t]，判断 Δt 与[t]的某种关系特征值 x，系统，决定奖励或者惩罚事务 T 的入口共识节点，奖励和惩罚的程度和 x 的值有关，具体可以自行设定；其中，奖金的来源可以是系统自动生成，也可以是一笔事务允诺的手续费。例如，当事务被引入共识绳后，系统生成一定量的奖金，或者收取所述事务的手续费作为奖金，或者系统同时生成一定量的奖金并收取所述事务的手续费作为奖金，再将所述奖金分配给所述事务的入口共识节点和参与该事务共识的共识节点，其中未诚实参与共识，例如，恶意逆转事务时序，恶意篡改事务内容的共识节点将无法获得所述事务的手续费，甚至系统会对未诚实参与共识的共识节点进行相应的处罚，例如，扣除押金、锁定共识节点的账户、暂停共识节点参与共识、永久清退共识节点等。

在一种实施方式中，一种无排序需求的事务的共识方法，包括：事务输入模块向共识节点输入无排序需求的事务；共识节点接收到所述事务后，将事务打结入本地到达绳，并为事务结打上共识时间戳后进行同步处理；共识节点按照共识时间的先后顺序对事务进行排序共识；所述无排序需求的事务，指的是在例如投票场景中，投票是一种无排序需求的事务，投票之间的先后顺序不影响投票结果。但是，为了避免“双花”问题，在竞选投票场景中可能出现一个人向两个人投票的情况，仍然需要对同一个投票人的事务进行排序，这时只需要利用事务的共识时间进行排序，把后写入的双花事务剔除掉即可。

由于没有排序需求，同一个应用场景的事务的手续费一般严格相同，共识节点对事务打时间戳也没有篡改时间的必要，而本申请所述的事务定序共识方法所形成的共识时间是网络内生时间，不依赖于外部时间源。因而可以直接用事务的共识时间戳进行排序。

让无排序需求的事务不参与有排序需求的事务的共识，这样可以降低有排序需求的事务的共识难度；同时无排序需求的事务可以利用有排序需求的事务共识后形成的共识时间作为时间戳，从而不需要对无排序需求的事务进行打包出块，也就降低了无排序需求的事务的共识难度。无排序需求的事务在创建时需要注明与有排序需求的事务的区别，例如：无排序需求的事务需要注明“无排序需求”，或者让有排序需求的事务注明“有排序需求”，或者事务输入模块通过专门的通道向共识节点输入无排序需求的事务，而共识节点有专门的模块处理

无排序需求的事务的共识。

无排序事务的应用场景一般会设置开始时间和结束时间，例如：投票场景中，组织方会设置投票开始时间和结束时间，方式是在开始时间向共识网络中输入一个“开始”事务，在结束时间向共识网络中输入一个“结束”事务，而所有投票人的投票事务的共识时间在开始事务和结束事务的共识时间之间就视为有效。可选的，可以要求共识节点对于在结束事务共识之后到达共识网络的事务不再做同步和共识处理，于是，对于有开始和结束需求的无排序事务应用场景，共识节点可以在到达事务的验证中加入对开始和结束时间的比对以决定是否对事务进行同步和共识处理的可选项。

在一种实施方式中，一种激励共识节点诚实进行共识处理的方法，包括：入口共识节点为接收的本地事务 T 加盖共识绳时间 t0；当事务 T 被写入到共识绳时，获取当前的共识绳时间 t1，则共识延迟 Δt=t1-t0；获取所述事务 T 临近的一组已共识事务的共识延迟的区间[t]，判断 Δt 与[t]的某种关系特征值 x，系统，决定奖励或者惩罚事务 T 的入口共识节点，奖励和惩罚的程度和 x 的值有关，具体可以自行设定；

其中，奖金的来源可以是系统自动生成，也可以是一笔事务允诺的手续费。例如，当事务被引入共识绳后，系统生成一定量的奖金，或者收取所述事务的手续费作为奖金，或者系统同时生成一定量的奖金并收取所述事务的手续费作为奖金，再将所述奖金分配给所述事务的入口共识节点和参与该事务共识的共识节点，其中未诚实参与共识（例如，恶意逆转事务时序，恶意篡改事务内容）的共识节点将无法获得所述事务的手续费，甚至系统会对未诚实参与共识的共识节点进行相应的处罚（例如，扣除押金、锁定共识节点的账户、暂停共识节点参与共识、永久清退共识节点等）。

在一种实施方式中，一种共识节点多级准入方法，其目的在于提高共识节点的忠诚度，以及提高共识网络安全性，包括：共识节点维护共识节点的身份状态数据。所述共识节点的身份状态包括：种子节点、正式节点、预备节点、培育节点，此外，节点可以临时作为巡视节点和介绍节点。其中：种子节点。一般用于最初的组网，并在共识网络中长期稳定运行，参与共识，可以获得奖励。正式节点。参与共识，可以作为入口共识节点，可以获得奖励，且奖励不被锁定。种子节点是一种特殊的正式节点。所有的正式节点需要每次向系统缴纳一定费用，所述费用将用于激励对共识网络的贡献者，例如开发人员。预备节点。参与共识，可以作为入口共识节点，可以获得奖励，但是奖励被锁定，当诚实参与共识的经历满足一定身份变更条件后，可以向正式节点发出身份变更申请，转为正式节点，称为正式节点后，已获得的奖励将解除锁定。所述奖励被锁定，指的是只能获取奖励而不能行驶奖励，例如获取到一笔钱，但是不能花这笔钱。培育节点。参与共识，可以作为入口共识节点，不能获得奖励，当诚实参与共识的经历满足一定身份变更条件后，可以向正式节点发出身份变更申请转

为预备节点。介绍节点。种子节点和正式节点可以作为介绍节点。当节点满足身份变更条件后，会按照预设的规则随机向多个（一般不少于 6 个）节点发出身份变更申请，接收到所述申请的节点被称为所述提出身份变更节点进行身份变更的介绍节点。介绍节点对所述身份变更申请进行审核，并将审核结果返回给所述提出所述身份变更申请的节点（简称申请节点），所述申请节点汇总到所有审核结果后（例如，必须全票通过，或者按照多数原则，或者三分之二通过）判断是否被最终审核通过，并将汇总结果作为一个事务同步到全网，所述事务的执行结果是对状态数据中的申请节点的身份进行变更或不变更。巡视节点。当共识节点满足预设条件时会获得一次巡视权利，所述巡视权利的行使指的是，共识节点随机向一个或多个共识节点输入一个或多个“巡视事务”，所述巡视事务同样将被共识处理写入共识绳。目的是利用巡视事务对逆转到达时序的行为进行破坏，另外巡视事务的入口共识节点要按照巡视事务的要求执行相应的验证。

在上述共识网络中，一个节点如果想加入共识网络，首先要经过介绍节点的介绍，成为培育节点，而后成为预备节点，最后成为正式节点，类似于入党介绍人、入党积极分子、预备党员和正式党员的概念。对上述共识节点多级准入方法中的身份状态进行变换名称、增减数量、变换职能，则不能视作创新，仍在本申请的保护范围内。

实施例 4

本提供一种共识应用程序，应用于所述一种事务定序共识系统，所述共识应用程序包括多个事务输入模块、多个状态获取模块和应用部署模块；所述应用部署模块，用于管理与共识应用相关的事务执行器和状态数据在共识节点的部署。如图 40 所示，所述事务定序共识系统 1 由 1 个共识网络和若干共识应用组成，其中，所述共识应用由若干个事务输入模块、若干个状态获取模块和应用部署模块组合而成，所述共识网络由共识节点组成。应用部署模块，管理本应用的可用节点信息，处理本应用部署资源匹配并部署或停止部署本应用，包括：可用节点管理子模块、部署资源匹配子模块。流程包括：

共识应用通过自身的事务输入模块向共识网络中的节点输入事务，客观上产生了“本真序”和各个节点的“到达序”；共识网络对共识应用输入的事务定序共识，各节点形成“还原序”，节点之间经过共识算法形成“共识序”，各共识节点按照共识序执行事务，使状态数据各副本产生一致的变化轨迹，将状态数据输出给共识应用；共识应用通过自身的状态获取模块从共识网络获取状态数据；

在本申请中，与共识应用相关的事务执行程序和状态数据需要部署在共识节点上，事务处理和状态处理均需要消耗共识节点的计算、存储和网络资源，共识应用需要为共识节点的资源付费，因此共识节点对不同共识应用的资源分配会影响到自身的收入，而共识应用对不同共识节点的部署会影响到自身的用户体验和支出成本。于是，共识节点和共识应用之间是

双向选择的关系。进一步的，在不同时间，共识应用的事务输出和状态获取的需求量可能不同，共识节点的计算和存储负载也可能不同，于是共识应用和共识节点的成本和收益处在动态变化中，共识应用需要实现可移动部署和可伸缩资源利用。

部署资源匹配子模块，完成本共识应用与共识节点的资源匹配和部署，实现共识应用的可移动、可伸缩部署。一方面，部署资源匹配子模块根据共识节点的资源报价选择意向共识节点，并向所述意向共识节点发送部署请求，共识节点根据共识应用的付费承诺和受欢迎程度等信息决定是否接受所述部署请求；另一方面，共识节点向共识应用发送部署邀请，共识应用的部署资源匹配子模块根据共识节点的情况判断是否接受所述部署邀请。当共识节点和共识应用互相匹配后，共识应用和共识节点会达成部署协议，共识节点会按照预设规则下载与已匹配共识应用相关的程序和数据，共识应用会将已匹配的共识节点信息在可用节点管理子模块备案。对于停止部署而言，当部署资源匹配子模块认为共识节点不适合继续部署本共识应用后，或者当共识节点认为共识应用不适合继续部署在本共识节点后，或者双方约定的部署时间到期后，双方可以按照部署协议停止部署。

可用节点管理子模块，管理共识节点信息。在本申请中，可用节点有两种含义，其中，对共识应用的资源匹配而言，可用节点是被本共识应用识别，正常运行且符合应用部署需求的共识节点集合。对共识应用的事务输入而言，可用节点是被本共识应用识别，正常运行且已部署本共识应用的事务处理程序和状态数据的共识节点集合。

在一种实施方式中，用户和共识应用之间是双向选择关系。一方面，用户根据自己的使用需求自主选择共识应用并在用户设备中安装共识应用的整体或部分模块，例如：用户 A 在自己手机里安装了电影订票应用，在电脑里安装了股票交易应用；另一方面，共识应用可以设定满足某种条件的用户才可以使用共识应用，例如，物品拍卖应用限制只有信用积分达到 500 的用户才可以进行物品竞拍活动。

本申请还提出一种专用于存储历史数据的设备。包括收发单元，至少用于收发已共识事务、已共识的共识绳和历史状态数据；处理单元，至少用于处理数据存储管理业务；存储单元，至少用于存储已共识事务、已共识的共识绳和历史状态数据；为了减轻执行节点的存储压力，可以设置专门的节点用于存储历史数据，例如：已执行的事务、已执行的共识绳、历史状态数据，可以称作“档案节点”。多个执行节点可以共享一个档案节点，共识节点和执行节点都可以监督档案节点，档案节点之间可以相互监督。档案节点只是进行增量式地数据存储，因而容易做到安全防护和数据恢复。

在本申请中，与共识应用相关的事务执行程序和状态数据需要部署在共识节点上，事务处理和状态处理均需要消耗共识节点的计算、存储和网络资源，共识应用需要为共识节点的资源付费，因此共识节点对不同共识应用的资源分配会影响到自身的收入，而共识应用对不

同共识节点的部署会影响到自身的用户体验和支出成本。于是，共识节点和共识应用之间是双向选择的关系。进一步的，在不同时间，共识应用的事务输出和状态获取的需求量可能不同，共识节点的计算和存储负载也可能不同，于是共识应用和共识节点的成本和收益处在动态变化中，共识应用需要实现可移动部署和可伸缩资源利用。

以上内容仅为说明本申请的技术思想，不能以此限定本申请的保护范围，凡是按照本申请提出的技术思想，在技术方案基础上所做的任何改动，均落入本申请权利要求书的保护范围之内。

此外，除非权利要求中明确说明，本申请所述处理元素和序列的顺序、数字字母的使用、或其他名称的使用，并非用于限定本申请流程和方法的顺序。尽管上述披露中通过各种示例讨论了一些目前认为有用的实施例，但应当理解的是，该类细节仅起到说明的目的，附加的权利要求并不仅限于披露的实施例，相反，权利要求旨在覆盖所有符合本申请实施例实质和范围的修正和等价组合。例如，虽然以上所描述的系统组件可以通过硬件设备实现，但是也可以只通过软件的解决方案得以实现，如在现有的服务器或移动设备上安装所描述的系统。

同理，应当注意的是，为了简化本申请披露的表述，从而帮助对一个或多个实施例的理解，前文对本申请实施例的描述中，有时会将多种特征归并至一个实施例、附图或对其的描述中。但是，这种披露方法并不意味着本申请对象所需要的特征比权利要求中提及的特征多。实际上，实施例的特征要少于上述披露的单个实施例的全部特征。

针对本申请引用的每个专利、专利申请、专利申请公开物和其他材料，如文章、书籍、说明书、出版物、文档等，特此将其全部内容并入本申请作为参考。与本申请内容不一致或产生冲突的申请历史文件除外，对本申请权利要求最广范围有限制的文件(当前或之后附加于本申请中的)也除外。需要说明的是，如果本申请附属材料中的描述、定义、和/或术语的使用与本申请所述内容有不一致或冲突的地方，以本申请的描述、定义和/或术语的使用为准。

说　明　书　附　图

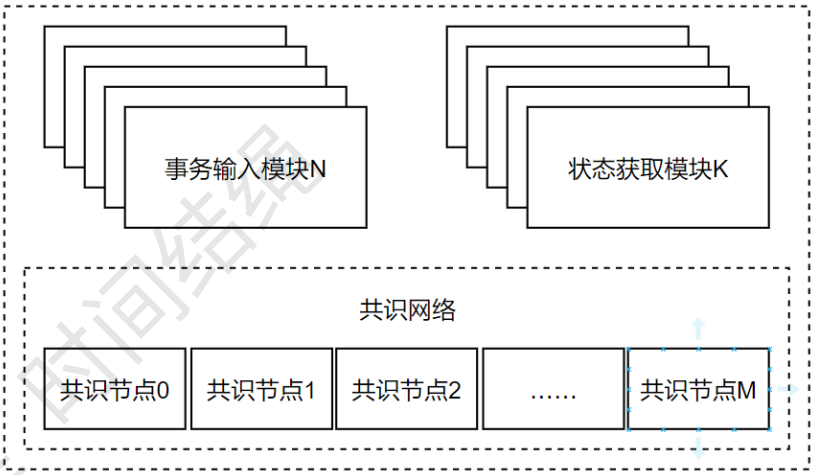

事务输入模块N
状态获取模块K
共识网络
共识节点0
共识节点1
共识节点2
……
共识节点M

图 1

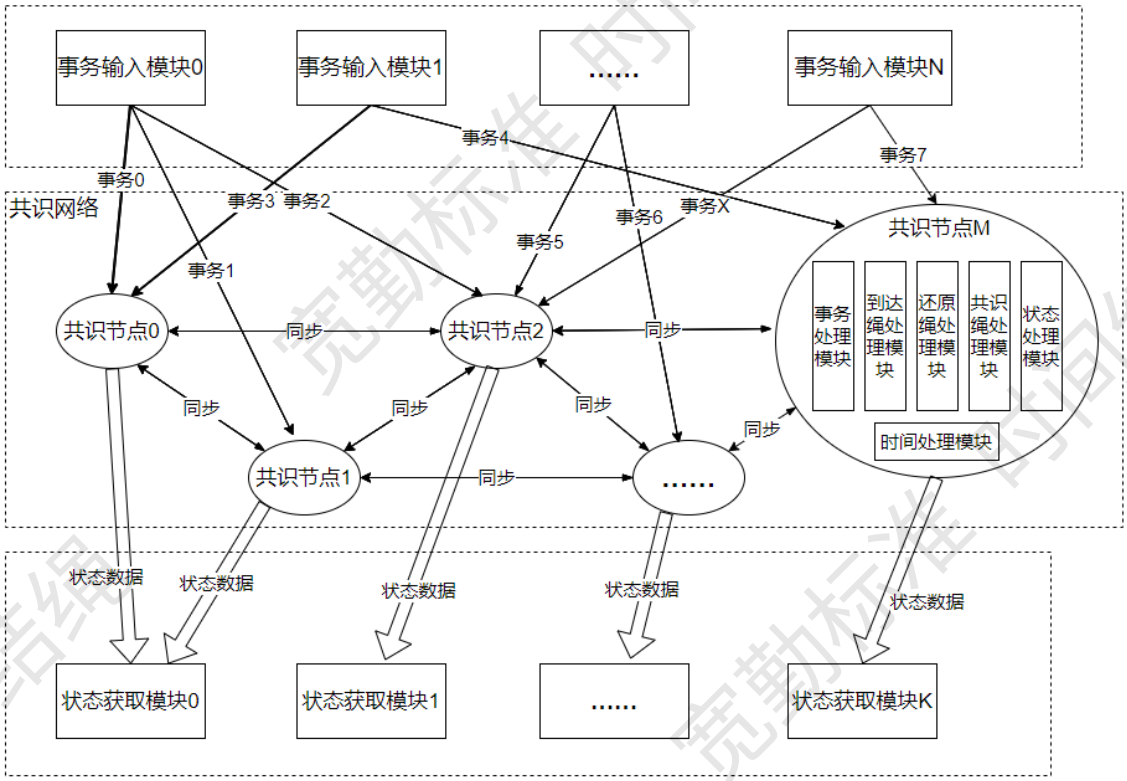

事务输入模块0
事务输入模块1
……
事务输入模块N
事务0
事务1
事务2
事务3
事务4
事务5
事务6
事务X
事务7
共识网络
共识节点0
共识节点1
共识节点2
……
共识节点M
同步
事务处理模块
到达绳处理模块
还原绳处理模块
共识绳处理模块
状态处理模块
时间处理模块
状态数据
状态获取模块0
状态获取模块1
……
状态获取模块K

图 2

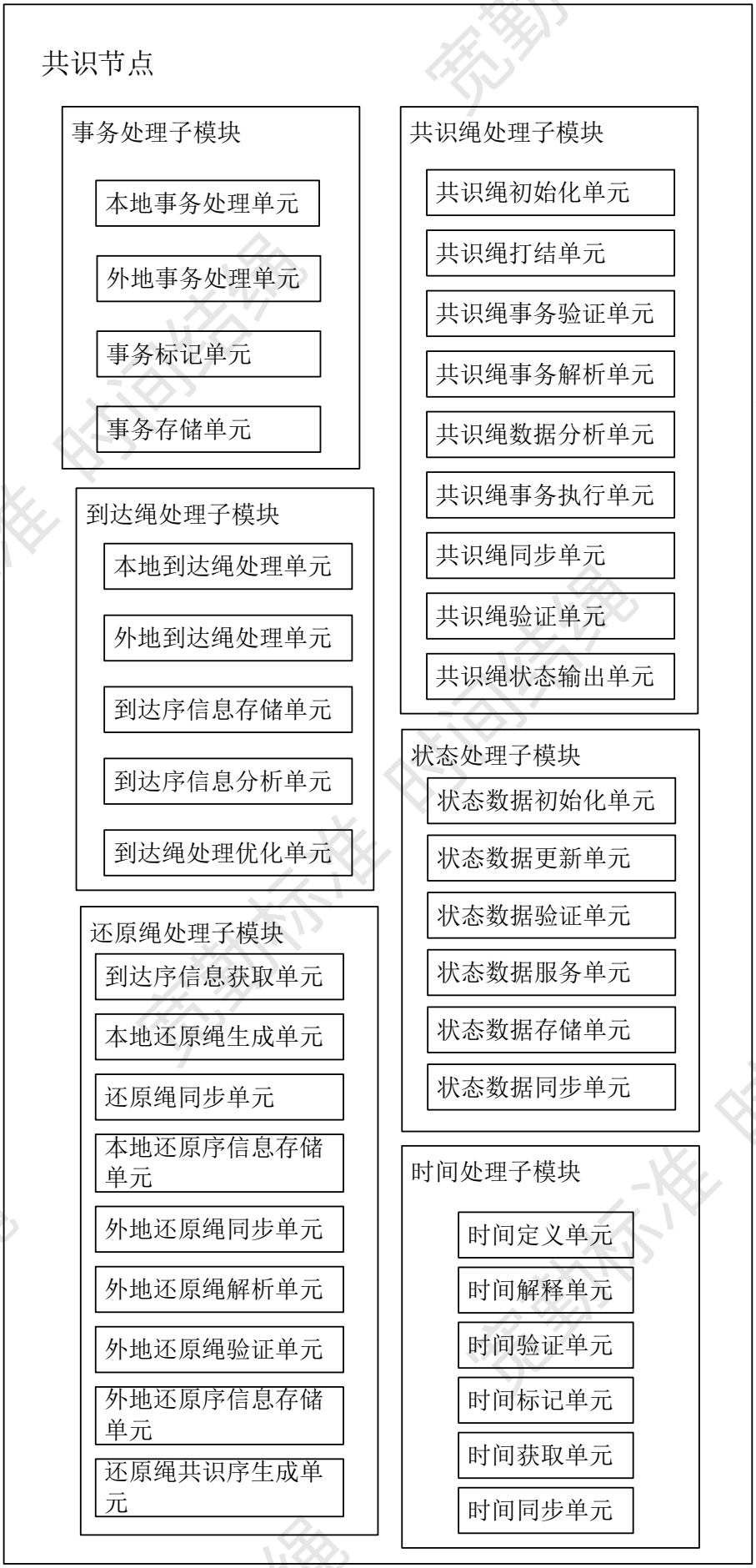

共识节点
事务处理子模块
本地事务处理单元
外地事务处理单元
事务标记单元
事务存储单元
到达绳处理子模块
本地到达绳处理单元
外地到达绳处理单元
到达序信息存储单元
到达序信息分析单元
到达绳处理优化单元
还原绳处理子模块
到达序信息获取单元
本地还原绳生成单元
还原绳同步单元
本地还原序信息存储单元
外地还原绳同步单元
外地还原绳解析单元
外地还原绳验证单元
外地还原序信息存储单元
还原绳共识序生成单元
共识绳处理子模块
共识绳初始化单元
共识绳打结单元
共识绳事务验证单元
共识绳事务解析单元
共识绳数据分析单元
共识绳事务执行单元
共识绳同步单元
共识绳验证单元
共识绳状态输出单元
状态处理子模块
状态数据初始化单元
状态数据更新单元
状态数据验证单元
状态数据服务单元
状态数据存储单元
状态数据同步单元
时间处理子模块
时间定义单元
时间解释单元
时间验证单元
时间标记单元
时间获取单元
时间同步单元

图 3

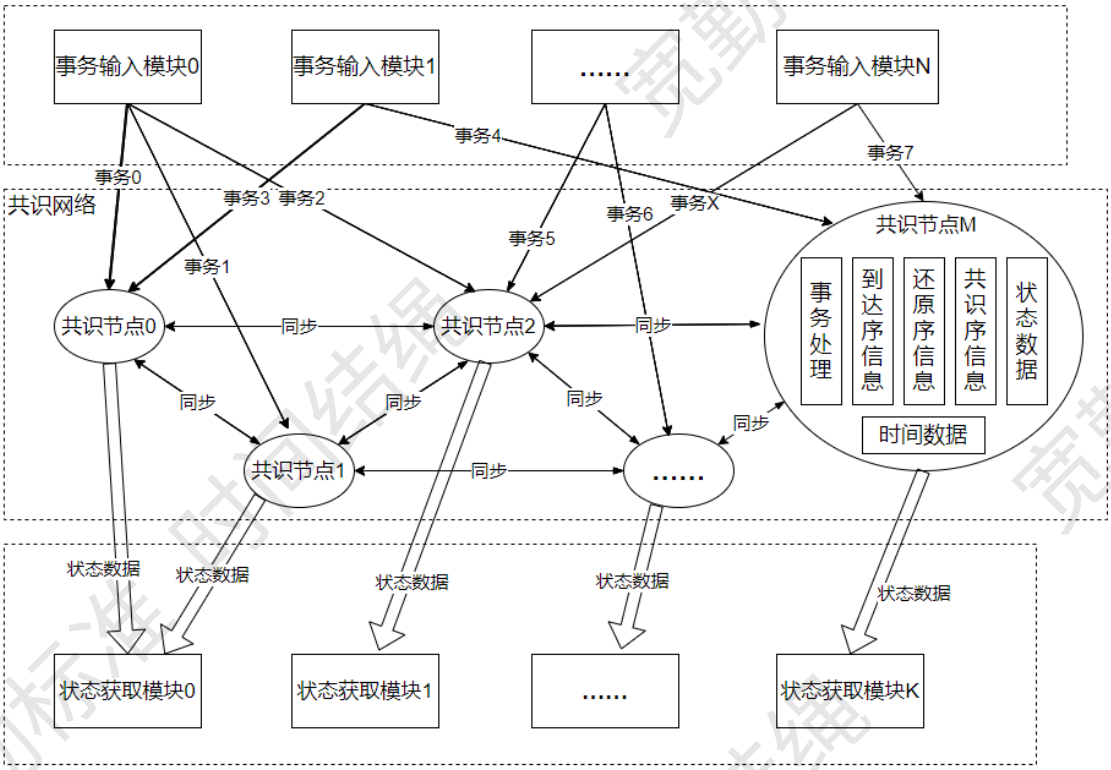

事务输入模块0
事务输入模块1
.....
事务输入模块N
事务4
事务7
事务0
共识网络
事务3
事务2
事务6
事务X
事务5
共识节点M
事务1
事务处理
到达序信息
还原序信息
共识序信息
状态数据
共识节点0
同步
共识节点2
同步
时间数据
同步
同步
同步
同步
共识节点1
同步
.....
状态数据
状态数据
状态数据
状态数据
状态数据
状态获取模块0
状态获取模块1
.....
状态获取模块K

图 4

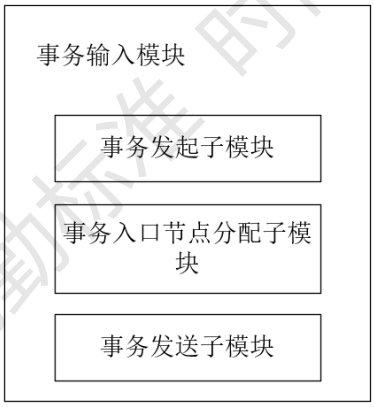

事务输入模块
事务发起子模块
事务入口节点分配子模块
事务发送子模块

图 5


状态获取模块
状态呈现子模块
状态合成子模块
状态请求子模块
状态存储子模块
状态订阅子模块
状态分析子模块

图 6

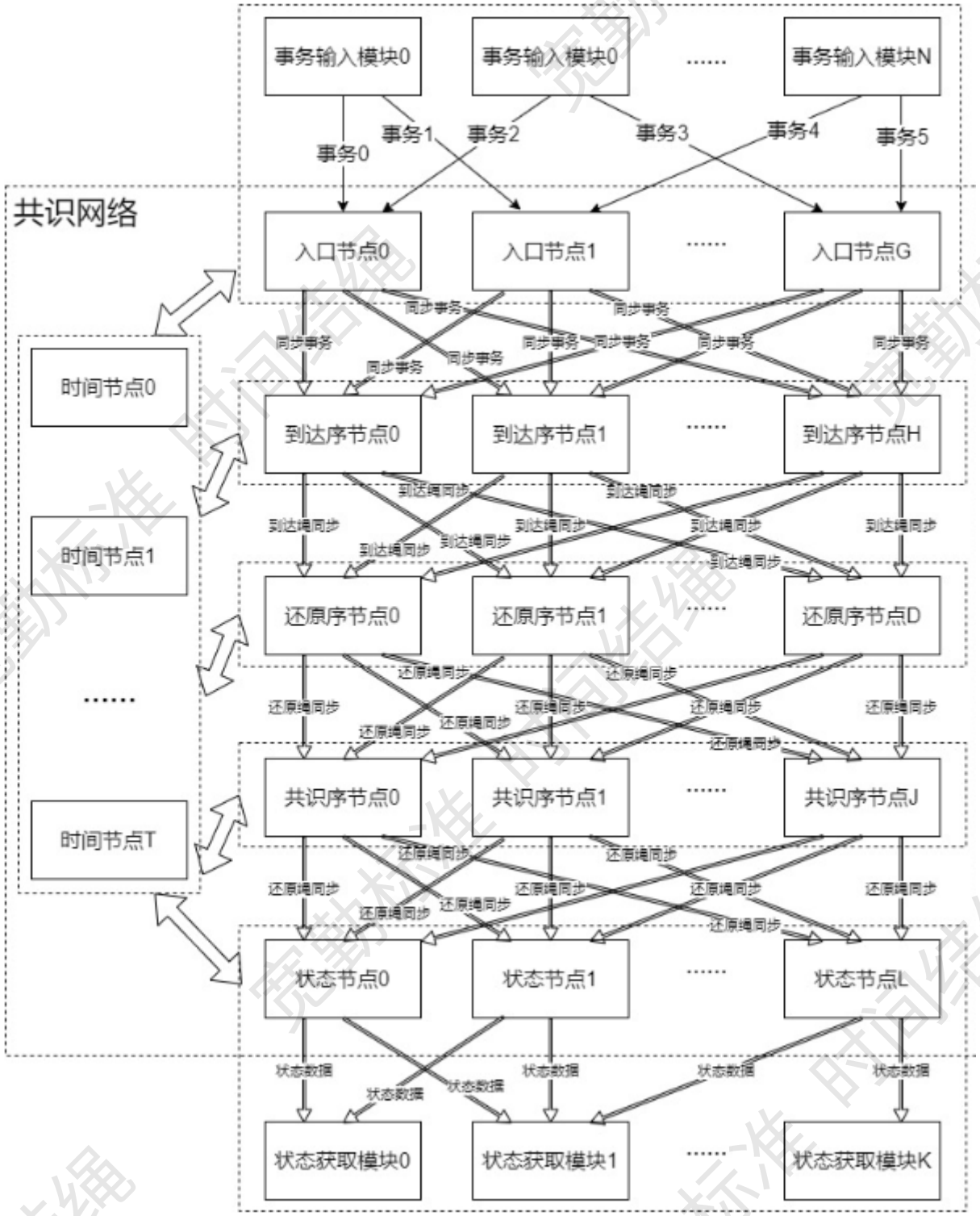

事务输入模块0
事务输入模块0
.....
事务输入模块N
事务0
事务1
事务2
事务3
事务4
事务5
共识网络
入口节点0
入口节点1
.....
入口节点G
同步事务
时间节点0
到达序节点0
到达序节点1
.....
到达序节点H
到达绳同步
时间节点1
还原序节点0
还原序节点1
.....
还原序节点D
还原绳同步
.....
共识序节点0
共识序节点1
.....
共识序节点J
时间节点T
状态节点0
状态节点1
.....
状态节点L
状态数据
状态获取模块0
状态获取模块1
.....
状态获取模块K

图 7

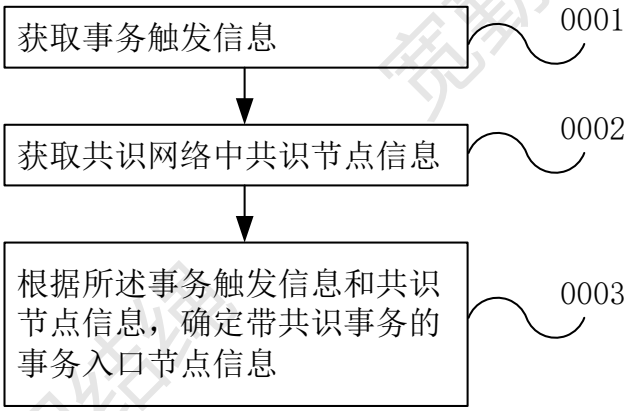

获取事务触发信息
0001
获取共识网络中共识节点信息
0002
根据所述事务触发信息和共识节点信息，确定带共识事务的事务入口节点信息
0003

图 8

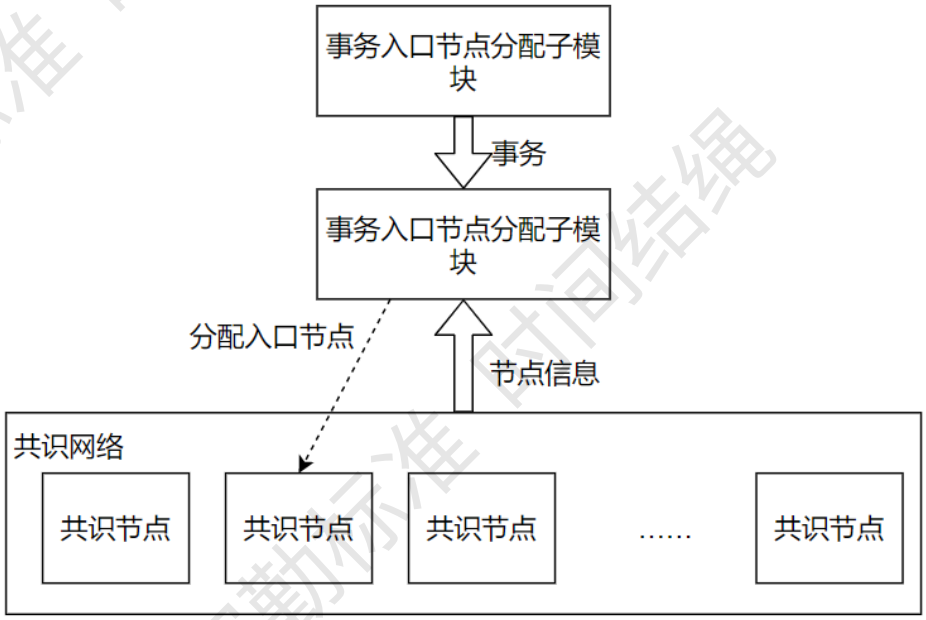

事务入口节点分配子模块
事务
事务入口节点分配子模块
分配入口节点
节点信息
共识网络
共识节点
共识节点
共识节点
.....
共识节点

图 9

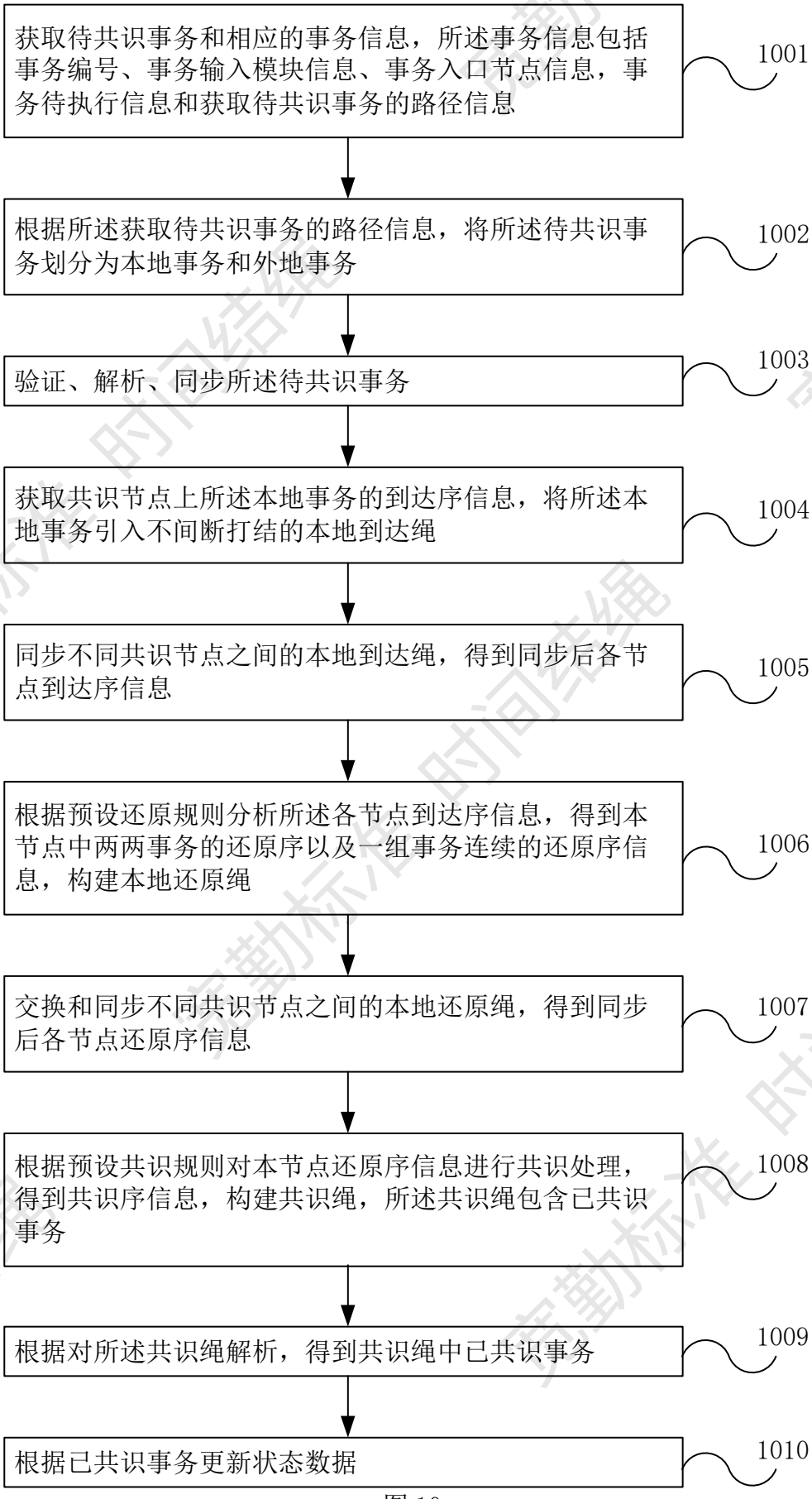

获取待共识事务和相应的事务信息，所述事务信息包括事务编号、事务输入模块信息、事务入口节点信息，事务待执行信息和获取待共识事务的路径信息
1001
根据所述获取待共识事务的路径信息，将所述待共识事务划分为本地事务和外地事务
1002
验证、解析、同步所述待共识事务
1003
获取共识节点上所述本地事务的到达序信息，将所述本地事务引入不间断打结的本地到达绳
1004
同步不同共识节点之间的本地到达绳，得到同步后各节点到达序信息
1005
根据预设还原规则分析所述各节点到达序信息，得到本节点中两两事务的还原序以及一组事务连续的还原序信息，构建本地还原绳
1006
交换和同步不同共识节点之间的本地还原绳，得到同步后各节点还原序信息
1007
根据预设共识规则对本节点还原序信息进行共识处理，得到共识序信息，构建共识绳，所述共识绳包含已共识事务
1008
根据对所述共识绳解析，得到共识绳中已共识事务
1009
根据已共识事务更新状态数据
1010

图 10

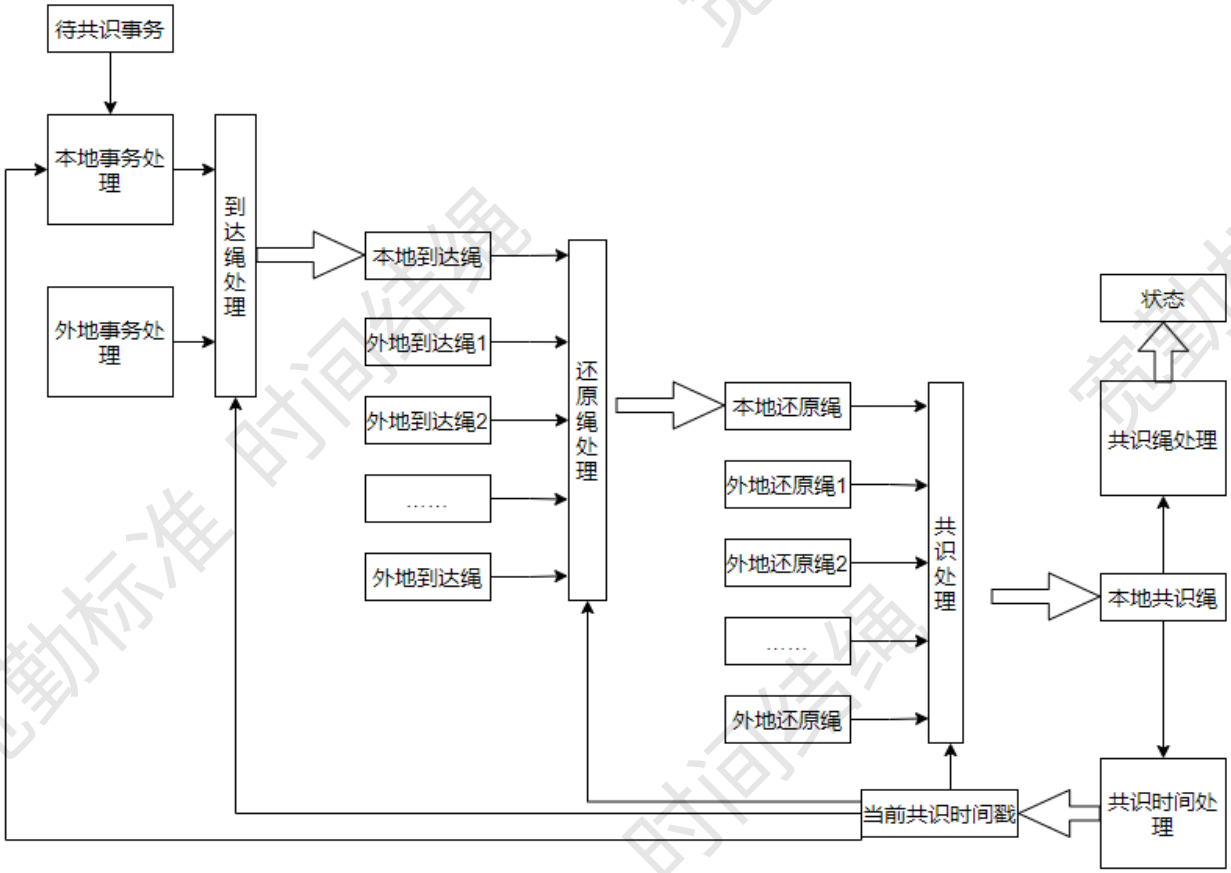


图 11

待共识事务
事务ID
入口节点ID
事务输入模块ID
事务本地时间戳
事务共识时间戳
事务输入模块签名
入口节点签名
事务待执行内容
事务执行器ID

图 12

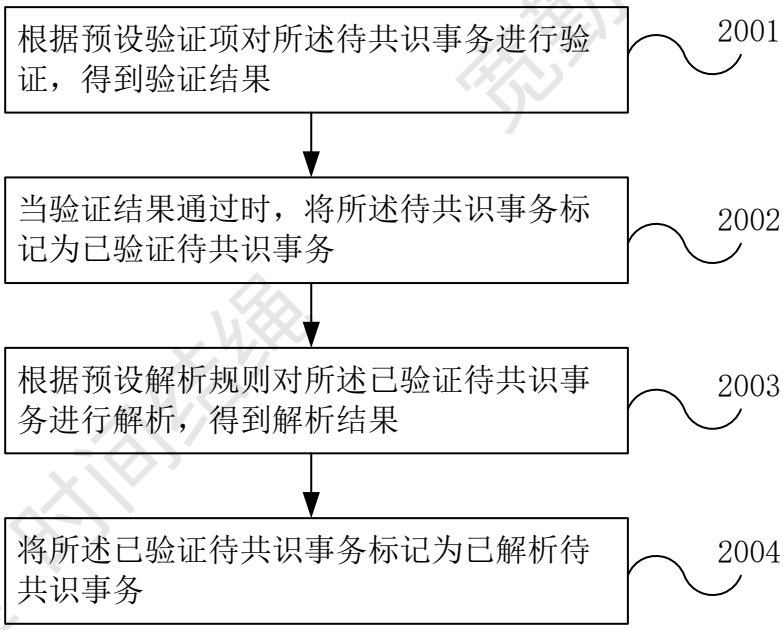

根据预设验证项对所述待共识事务进行验证，得到验证结果
2001
当验证结果通过时，将所述待共识事务标记为已验证待共识事务
2002
根据预设解析规则对所述已验证待共识事务进行解析，得到解析结果
2003
将所述已验证待共识事务标记为已解析待共识事务
2004

图 13

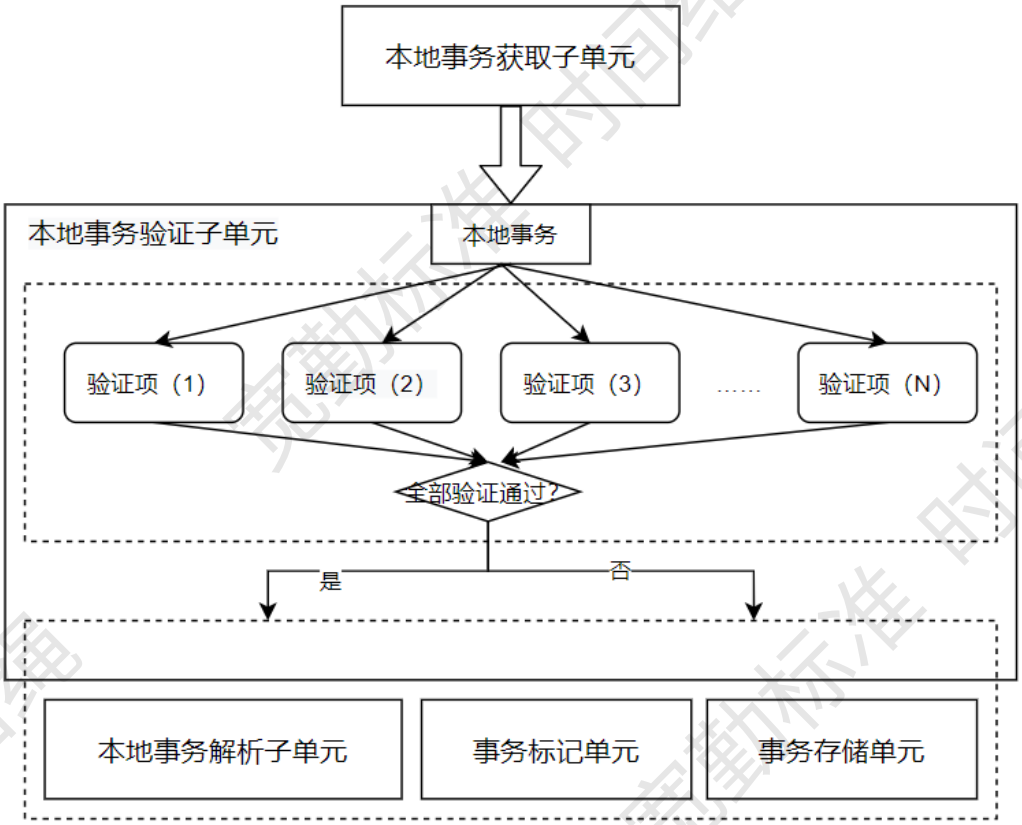

本地事务获取子单元
本地事务验证子单元
本地事务
验证项（1）
验证项（2）
验证项（3）
.....
验证项（N）
全部验证通过？
是
否
本地事务解析子单元
事务标记单元
事务存储单元

图 14

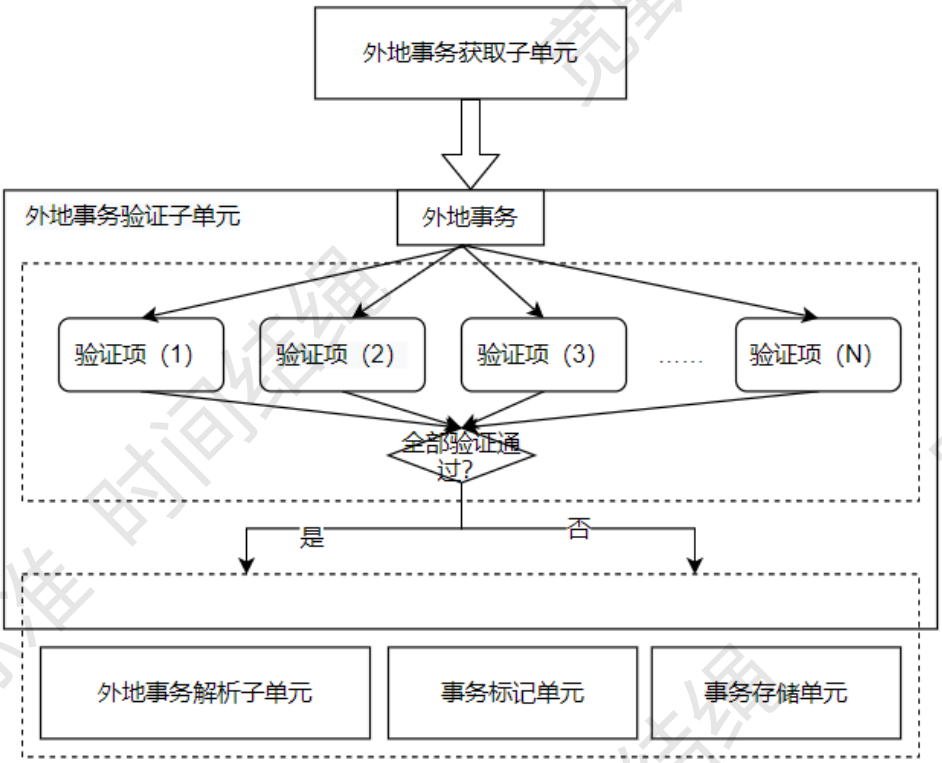

外地事务获取子单元
外地事务验证子单元
外地事务
验证项（1）
验证项（2）
验证项（3）
……
验证项（N）
全部验证通过?
是
否
外地事务解析子单元
事务标记单元
事务存储单元

图 15

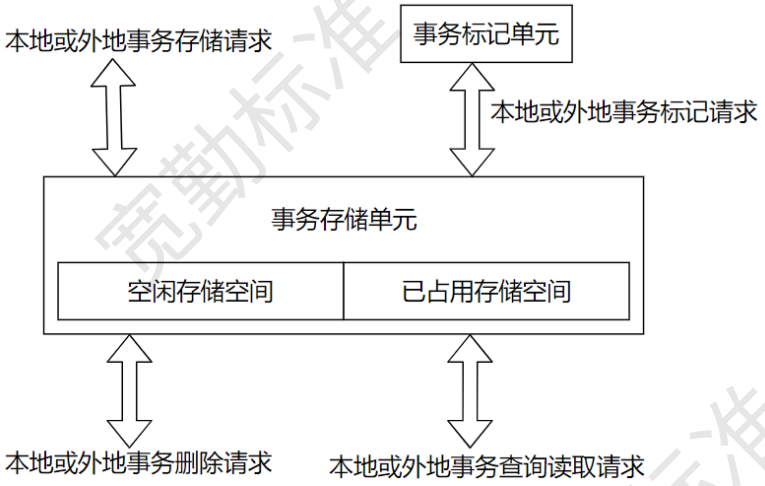

本地或外地事务存储请求
事务标记单元
本地或外地事务标记请求
事务存储单元
空闲存储空间
已占用存储空间
本地或外地事务删除请求
本地或外地事务查询读取请求

图 16

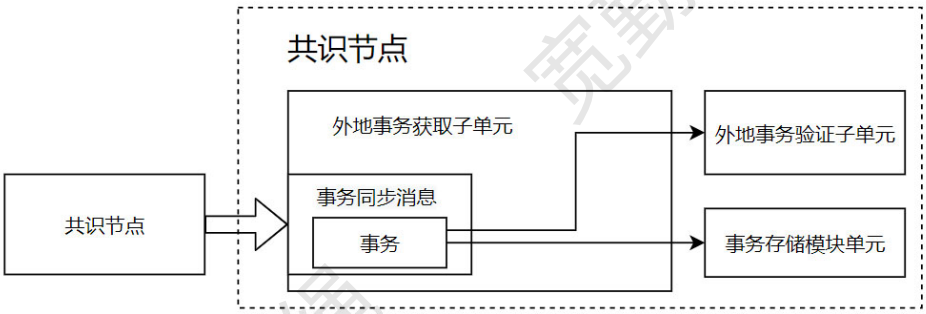

共识节点
外地事务获取子单元
外地事务验证子单元
事务同步消息
事务
共识节点
事务存储模块单元

图 17

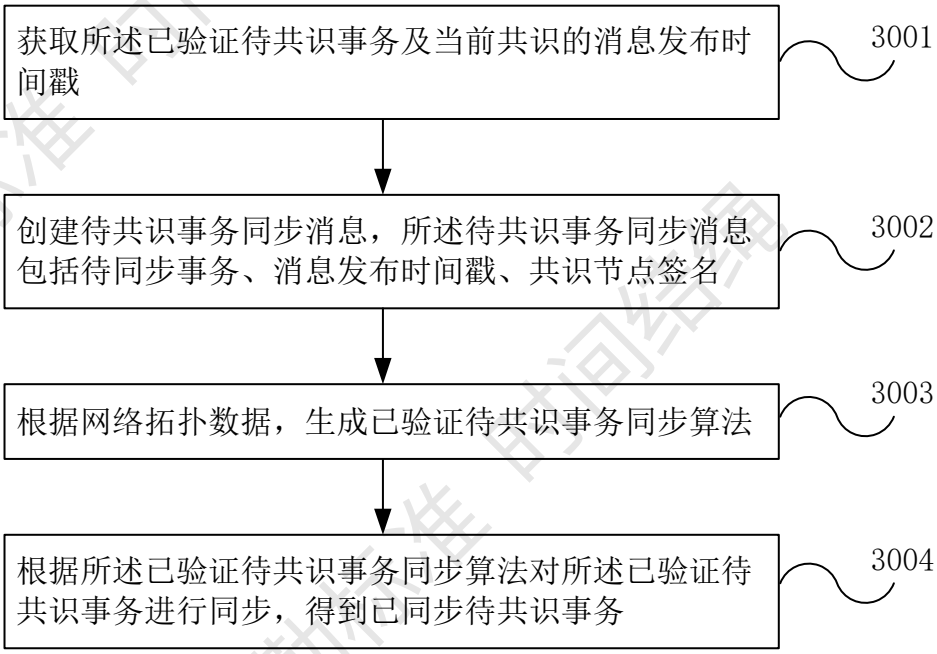

获取所述已验证待共识事务及当前共识的消息发布时间戳
3001
创建待共识事务同步消息，所述待共识事务同步消息包括待同步事务、消息发布时间戳、共识节点签名
3002
根据网络拓扑数据，生成已验证待共识事务同步算法
3003
根据所述已验证待共识事务同步算法对所述已验证待共识事务进行同步，得到已同步待共识事务
3004

图 18

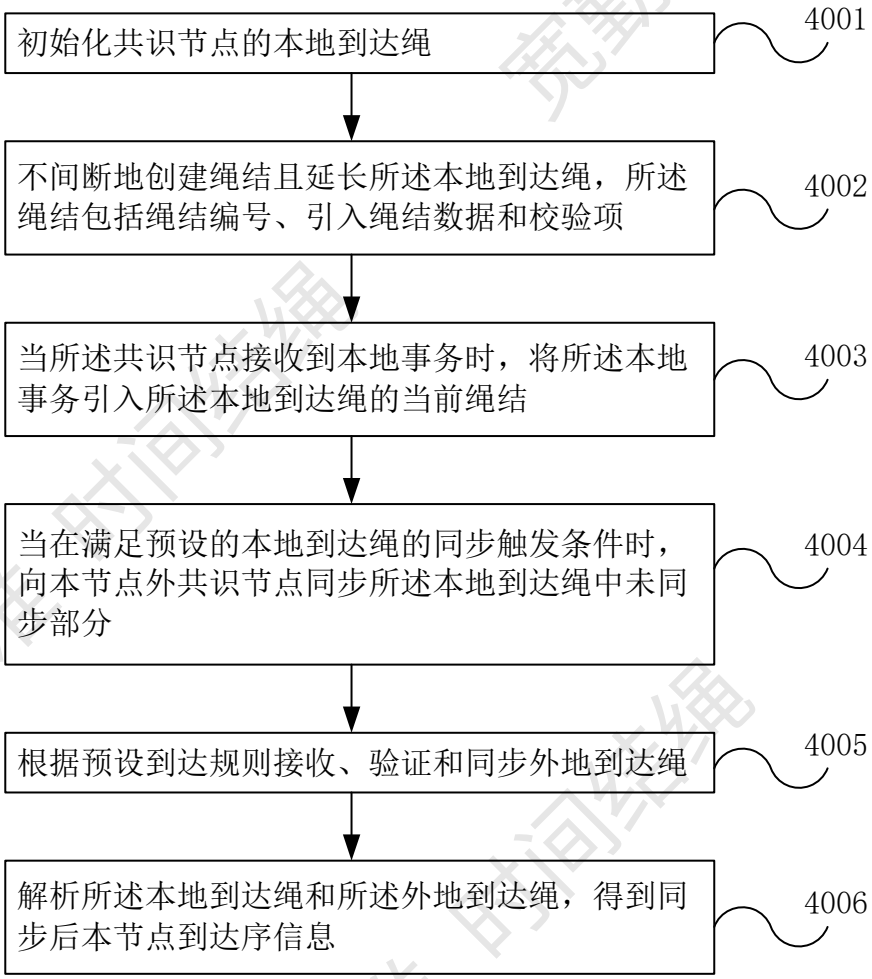


图 19

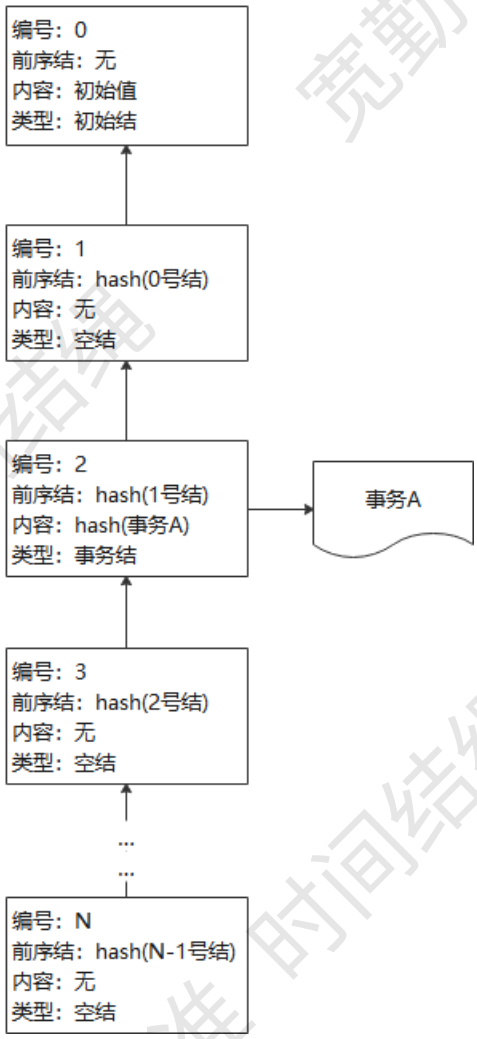

编号：0
前序结：无
内容：初始值
类型：初始结
编号：1
前序结：hash(0号结)
内容：无
类型：空结
编号：2
前序结：hash(1号结)
内容：hash(事务A)
类型：事务结
事务A
编号：3
前序结：hash(2号结)
内容：无
类型：空结
...
...
编号：N
前序结：hash(N-1号结)
内容：无
类型：空结

图 20


绳结
绳结编号
前序结引用
事务引用
校验项
校验项
校验规则编号
校验规则编号

图 21

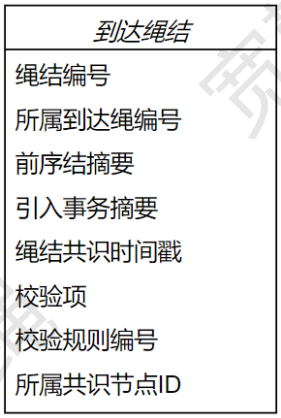

到达绳结
绳结编号
所属到达绳编号
前序结摘要
引入事务摘要
绳结共识时间戳
校验项
校验规则编号
所属共识节点ID

图 22

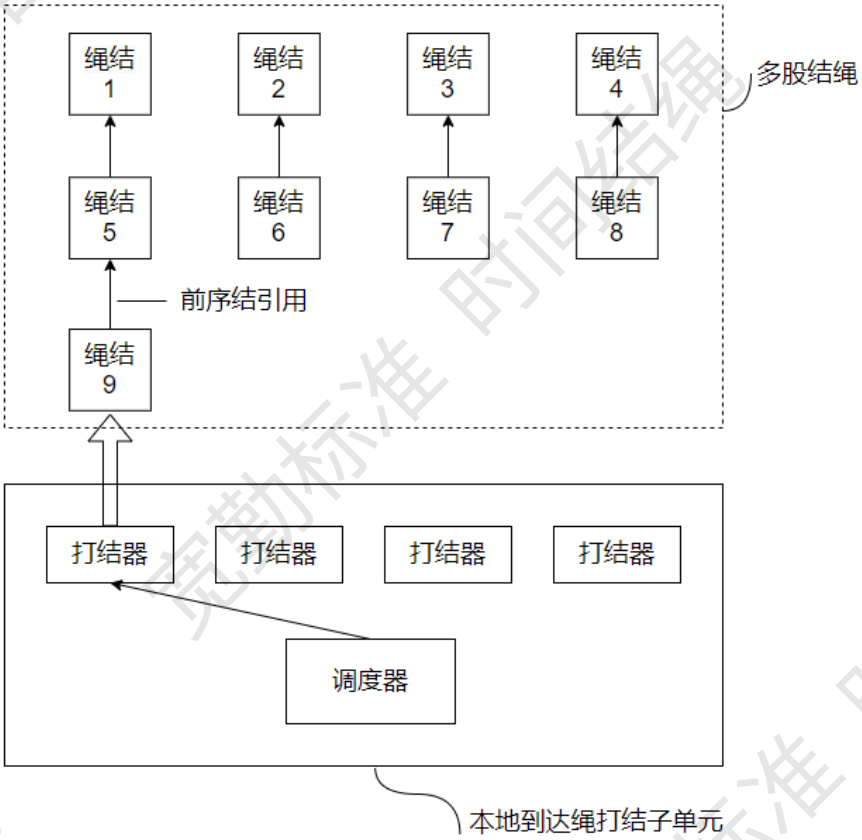

绳结 1
绳结 2
绳结 3
绳结 4
多股结绳
绳结 5
绳结 6
绳结 7
绳结 8
前序结引用
绳结 9
打结器
打结器
打结器
打结器
调度器
本地到达绳打结子单元

图 23

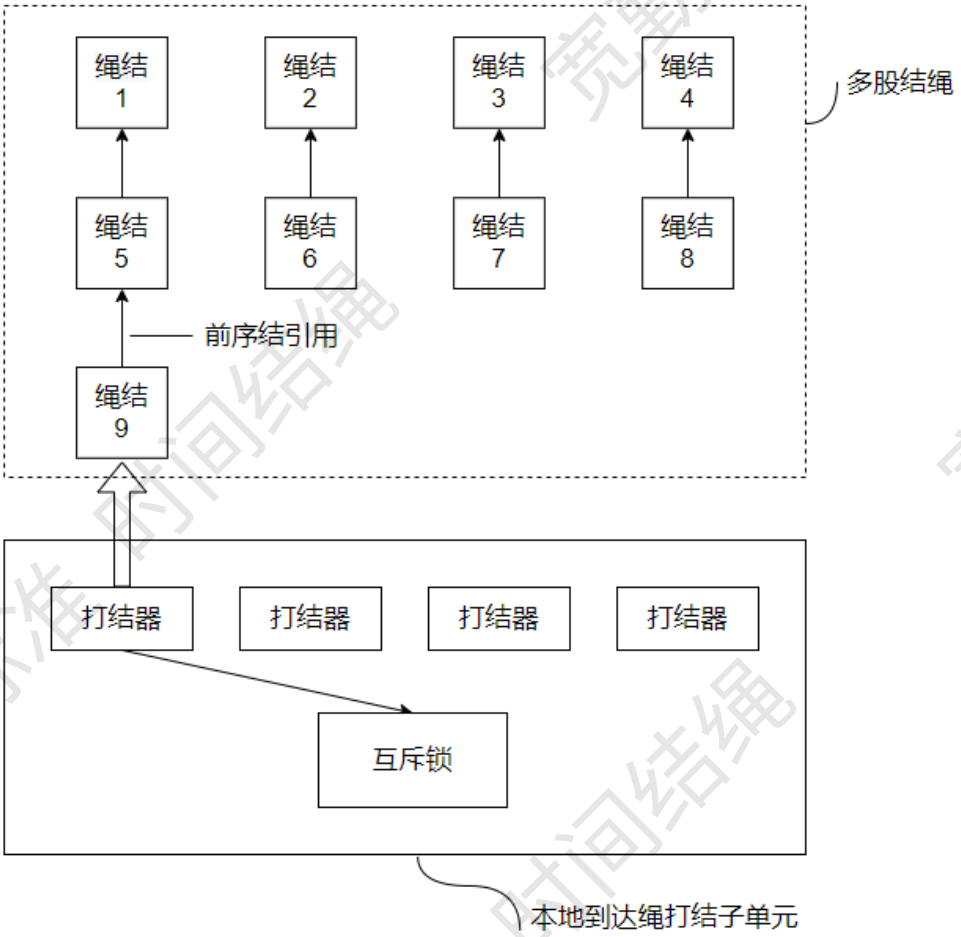

绳结1
绳结2
绳结3
绳结4
多股结绳
绳结5
绳结6
绳结7
绳结8
前序结引用
绳结9
打结器
打结器
打结器
打结器
互斥锁
本地到达绳打结子单元

图 24

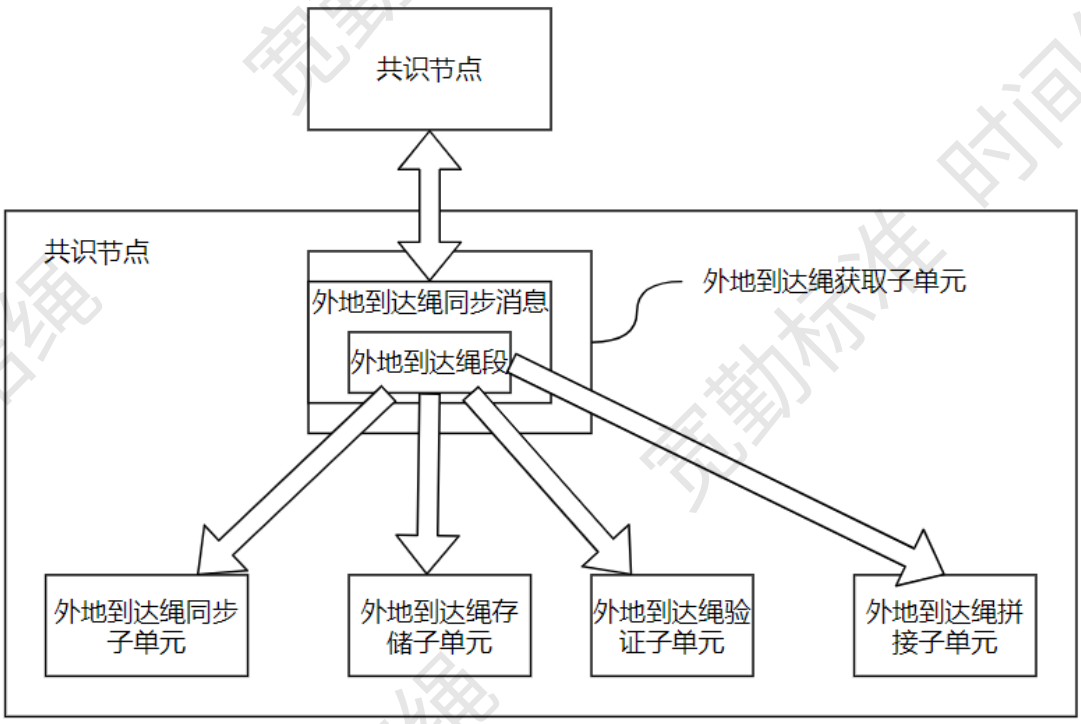

共识节点
共识节点
外地到达绳同步消息
外地到达绳段
外地到达绳获取子单元
外地到达绳同步子单元
外地到达绳存储子单元
外地到达绳验证子单元
外地到达绳拼接子单元

图 25

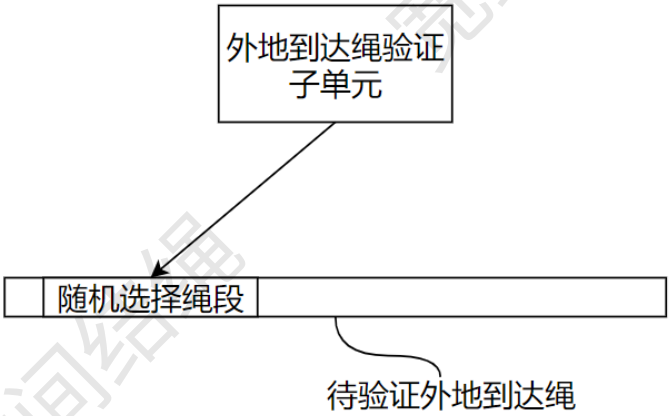

外地到达绳验证子单元
随机选择绳段
待验证外地到达绳

图 26

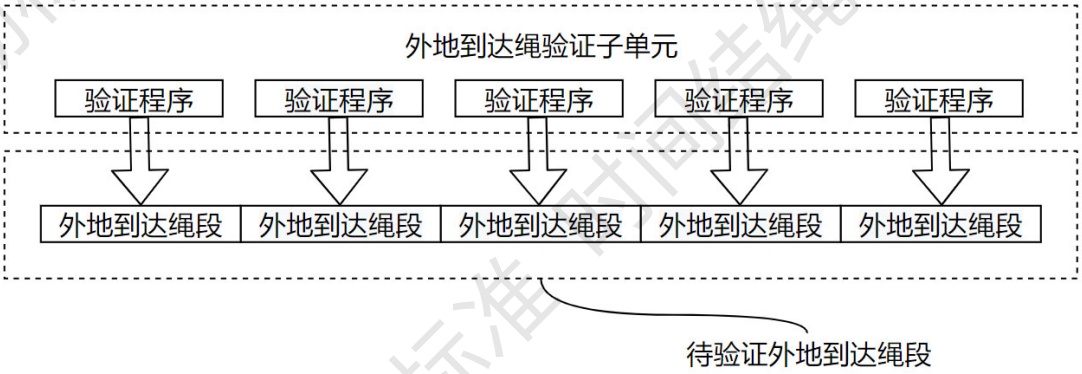

外地到达绳验证子单元
验证程序
验证程序
验证程序
验证程序
验证程序
外地到达绳段
外地到达绳段
外地到达绳段
外地到达绳段
外地到达绳段
待验证外地到达绳段

图 27

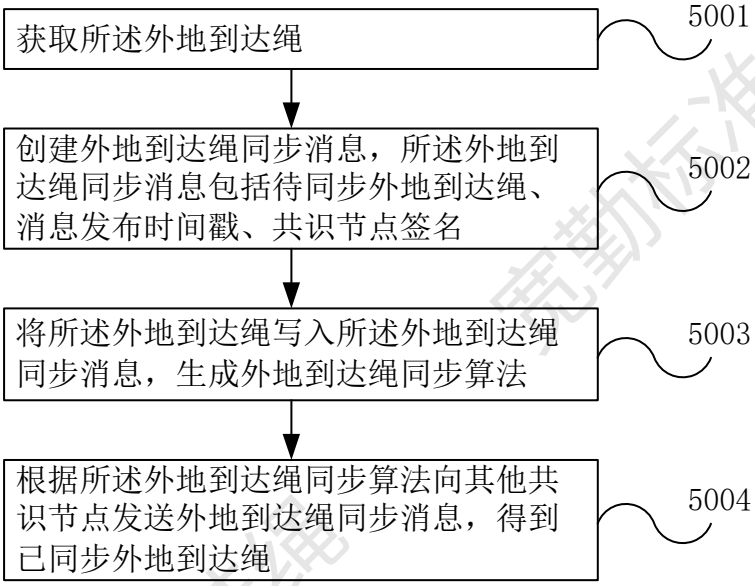

获取所述外地到达绳
5001
创建外地到达绳同步消息，所述外地到达绳同步消息包括待同步外地到达绳、消息发布时间戳、共识节点签名
5002
将所述外地到达绳写入所述外地到达绳同步消息，生成外地到达绳同步算法
5003
根据所述外地到达绳同步算法向其他共识节点发送外地到达绳同步消息，得到已同步外地到达绳
5004

图 28

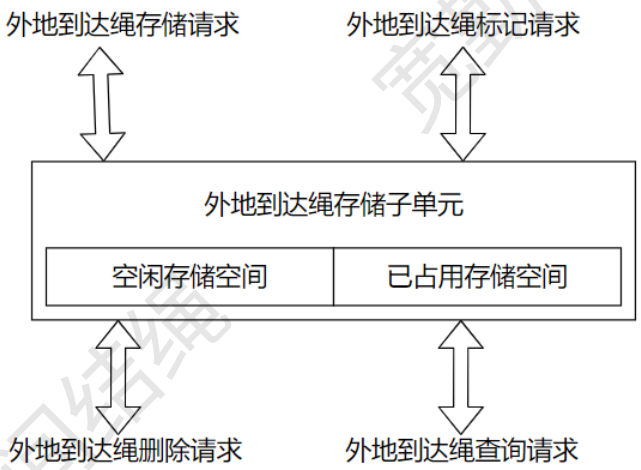

外地到达绳存储请求
外地到达绳标记请求
外地到达绳存储子单元
空闲存储空间
已占用存储空间
外地到达绳删除请求
外地到达绳查询请求

图 29

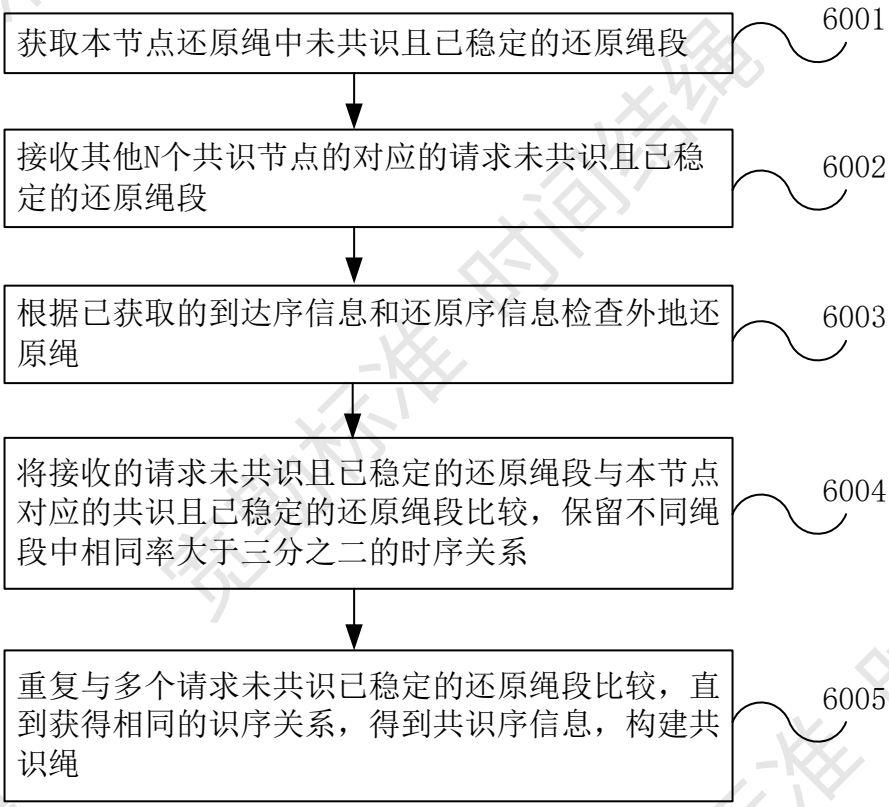

获取本节点还原绳中未共识且已稳定的还原绳段
6001
接收其他N个共识节点的对应的请求未共识且已稳定的还原绳段
6002
根据已获取的到达序信息和还原序信息检查外地还原绳
6003
将接收的请求未共识且已稳定的还原绳段与本节点对应的共识且已稳定的还原绳段比较，保留不同绳段中相同率大于三分之二的时序关系
6004
重复与多个请求未共识已稳定的还原绳段比较，直到获得相同的识序关系，得到共识序信息，构建共识绳
6005

图 30

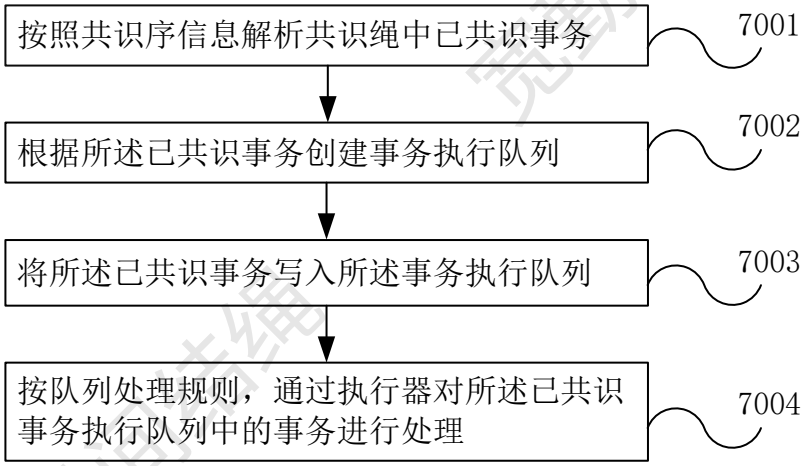

按照共识序信息解析共识绳中已共识事务
7001
根据所述已共识事务创建事务执行队列
7002
将所述已共识事务写入所述事务执行队列
7003
按队列处理规则，通过执行器对所述已共识事务执行队列中的事务进行处理
7004

图 31

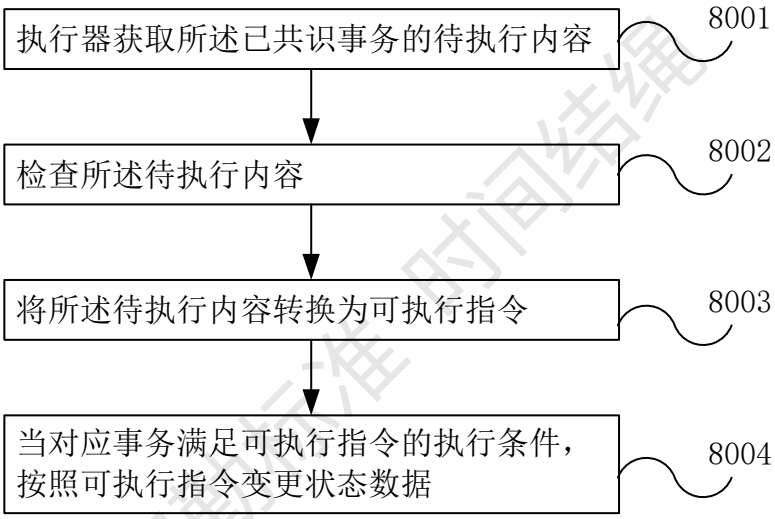

执行器获取所述已共识事务的待执行内容
8001
检查所述待执行内容
8002
将所述待执行内容转换为可执行指令
8003
当对应事务满足可执行指令的执行条件，按照可执行指令变更状态数据
8004

图 32

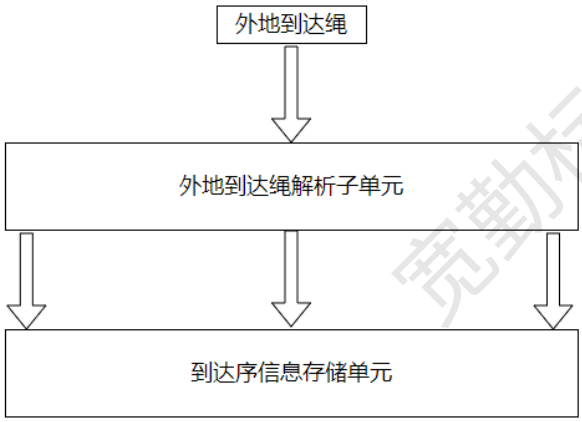

外地到达绳
外地到达绳解析子单元
到达序信息存储单元

图 33

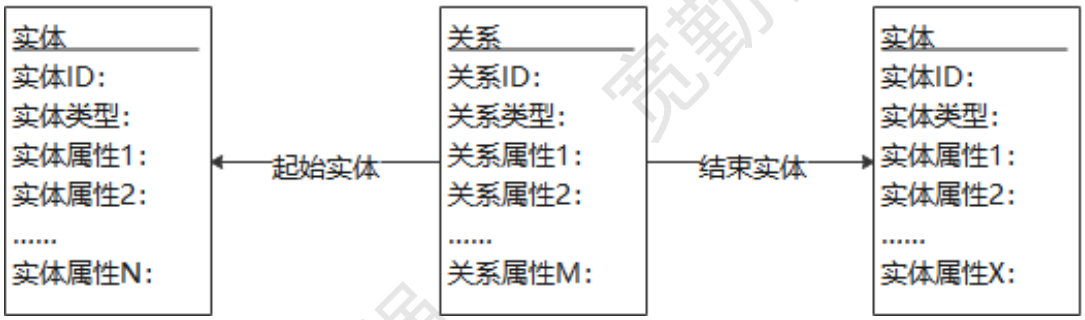

实体
实体ID:
实体类型:
实体属性1:
实体属性2:
.....
实体属性N:
起始实体
关系
关系ID:
关系类型:
关系属性1:
关系属性2:
.....
关系属性M:
结束实体
实体
实体ID:
实体类型:
实体属性1:
实体属性2:
.....
实体属性X:

图 34

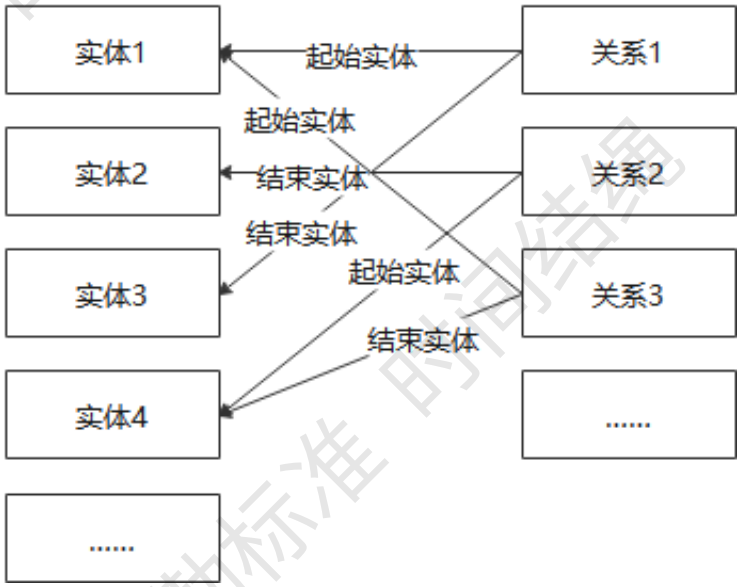

实体1
起始实体
关系1
起始实体
实体2
结束实体
关系2
结束实体
实体3
起始实体
关系3
结束实体
实体4
.....
.....

图 35

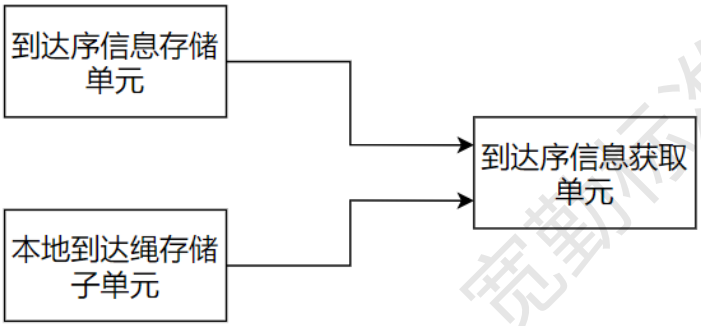

到达序信息存储单元
到达序信息获取单元
本地到达绳存储子单元

图 36

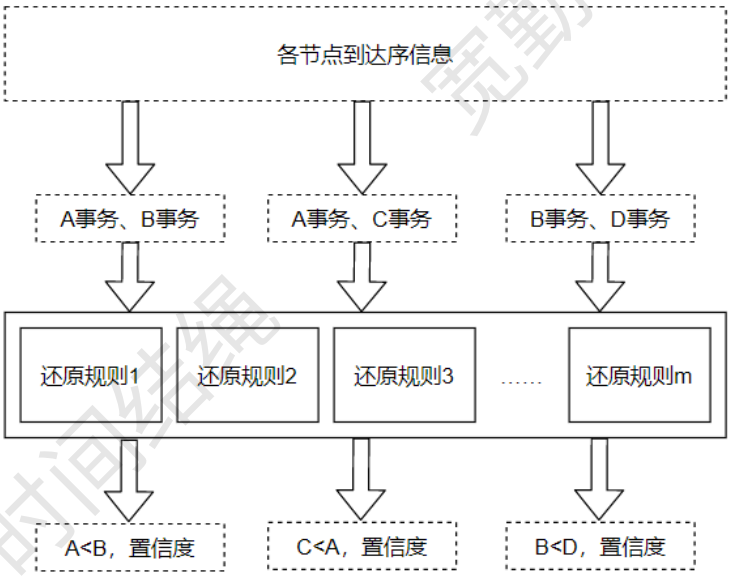

各节点到达序信息
A事务、B事务
A事务、C事务
B事务、D事务
还原规则1
还原规则2
还原规则3
.....
还原规则m
A<B，置信度
C<A，置信度
B<D，置信度

图 37

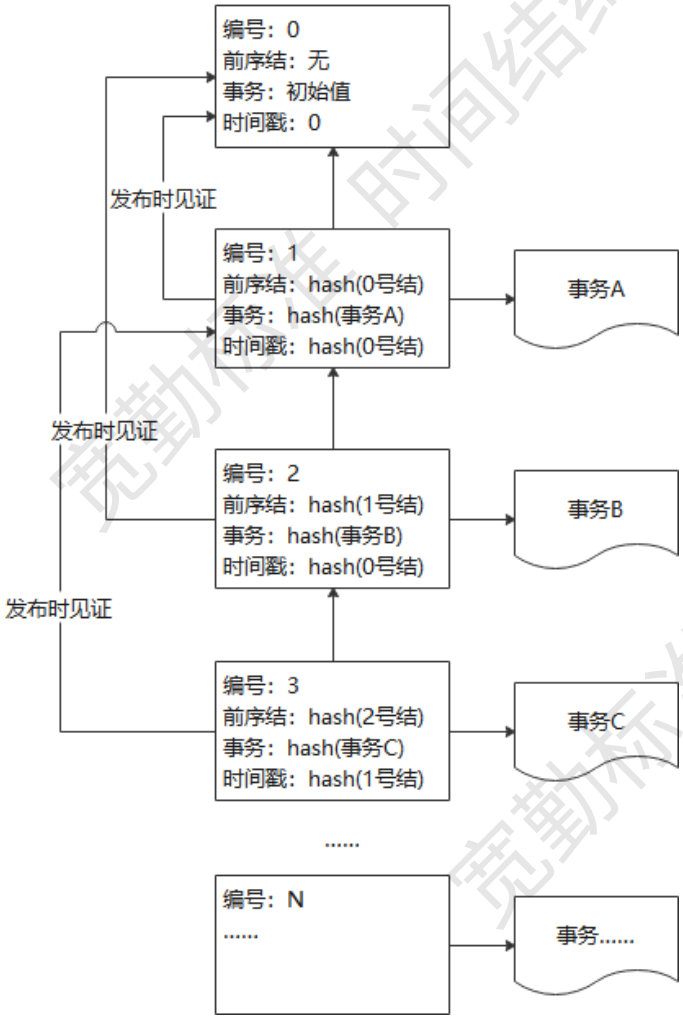

编号：0
前序结：无
事务：初始值
时间戳：0
发布时见证
编号：1
前序结：hash(0号结)
事务：hash(事务A)
时间戳：hash(0号结)
事务A
发布时见证
编号：2
前序结：hash(1号结)
事务：hash(事务B)
时间戳：hash(0号结)
事务B
发布时见证
编号：3
前序结：hash(2号结)
事务：hash(事务C)
时间戳：hash(1号结)
事务C
.....
编号：N
.....
事务.....

图 38

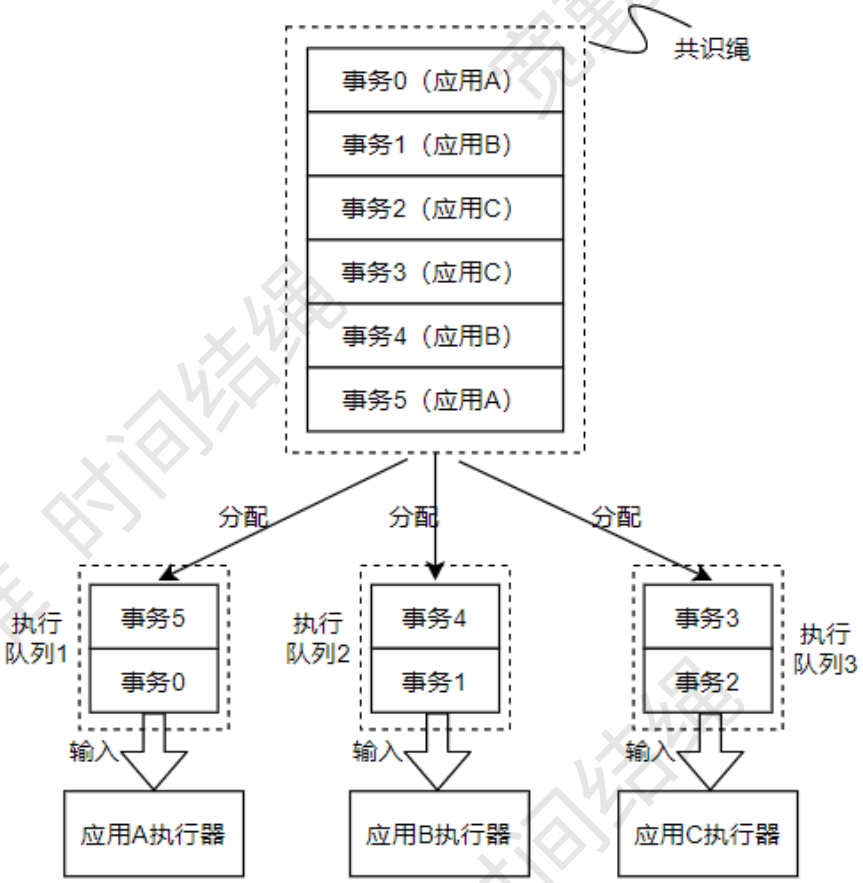

共识绳
事务0（应用A）
事务1（应用B）
事务2（应用C）
事务3（应用C）
事务4（应用B）
事务5（应用A）
分配
分配
分配
执行队列1
事务5
事务0
执行队列2
事务4
事务1
执行队列3
事务3
事务2
输入
输入
输入
应用A执行器
应用B执行器
应用C执行器

图 39

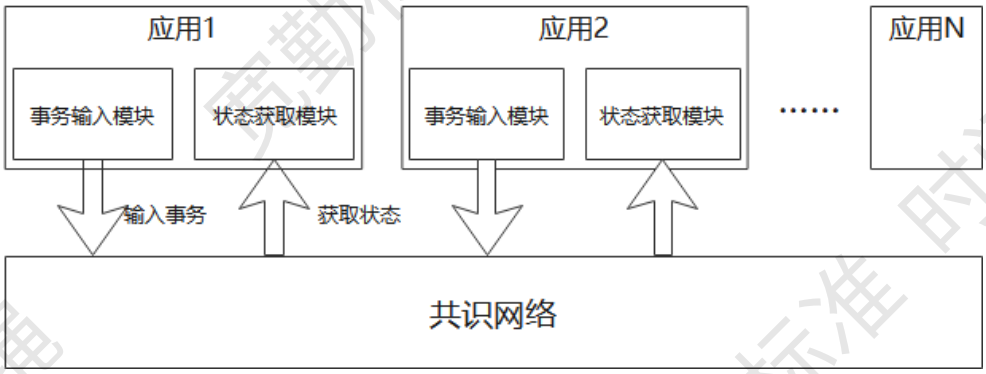

应用1
事务输入模块
状态获取模块
应用2
事务输入模块
状态获取模块
.....
应用N
输入事务
获取状态
共识网络

图 40