(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 111679978 A
(43)申请公布日 2020.09.18

(21)申请号 202010482333.8

(22)申请日 2020.05.29

(71)申请人 腾讯科技(深圳)有限公司
地址 518057 广东省深圳市南山区高新区科技中一路腾讯大厦35层

(72)发明人 陈金龙

(74)专利代理机构 广州三环专利商标代理有限公司 44202
代理人 熊永强 杜维

(51)Int.Cl.
G06F 11/36(2006.01)

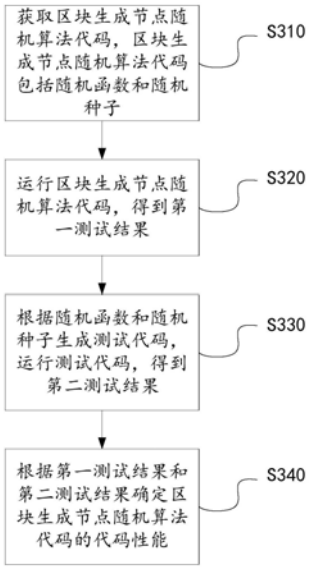
权利要求书2页 说明书13页 附图7页

(54)发明名称

一种程序测试方法、程序测试装置、电子设备及存储介质

(57)摘要

本发明实施例提出了一种程序测试方法、程序测试装置、电子设备及存储介质。其中,程序测试方法具体是:获取区块生成节点随机算法程序,区块生成节点随机算法程序包括随机函数和随机种子;运行区块生成节点随机算法程序,得到第一测试结果;根据随机函数和随机种子生成测试程序,运行测试程序,得到第二测试结果;根据第一测试结果和第二测试结果确定区块生成节点随机算法程序的程序性能。通过本发明实施例所提供的程序测试方法,采用蜕变测试思想,引入自动化框架,自动挖掘区块链项目程序中的随机函数和随机种子,自动生成相应的测试程序,解决了区块生成节点选择算法随机性的测试预测问题,从而蜕变测试节点选择算法的随机性和可靠性。



CN 111679978 A

1.一种程序测试方法,其特征在于,所述方法包括:

获取区块生成节点随机算法程序,所述区块生成节点随机算法程序包括随机函数和随机种子;

运行所述区块生成节点随机算法程序,得到第一测试结果;

根据所述随机函数和所述随机种子生成测试程序,运行所述测试程序,得到第二测试结果;

根据所述第一测试结果和所述第二测试结果确定所述区块生成节点随机算法程序的程序性能。

2.根据权利要求1所述的方法,其特征在于,所述根据所述随机函数和所述随机种子生成测试程序,包括:

获取与所述区块生成节点随机算法程序对应的测试脚本;

将所述随机函数和所述随机种子添加至所述测试脚本;

对添加所述随机函数和所述随机种子后的测试脚本进行模板引擎渲染处理,得到所述测试程序。

3.根据权利要求1所述的方法,其特征在于,所述程序性能包括随机准确程度;

所述根据所述第一测试结果和所述第二测试结果确定所述区块生成节点随机算法程序的程序性能,包括:

统计所述第一测试结果和所述第二测试结果的结果匹配度;

根据所述结果匹配度确定所述随机准确程度;

将所述随机准确程度、所述第一测试结果以及所述第二测试结果组合为测试报告,输出所述测试报告。

4.根据权利要求3所述的方法,其特征在于,所述第二测试结果包括N个单位测试结果,N是正整数,所述N个单位测试结果是运行N次所述测试程序后所输出的测试结果;

所述统计所述第一测试结果和所述第二测试结果的结果匹配度,包括:

统计所述第一测试结果和所述N个单位测试结果的N个单位匹配度;

根据所述N个单位匹配度确定所述结果匹配度。

5.根据权利要求4所述的方法,其特征在于,还包括:

当所述随机准确程度大于预设程度阈值时,对所述第一测试结果进行哈希运算,得到所述第一测试结果的哈希值;

根据所述第一测试结果的哈希值,确定区块生成节点;所述区块生成节点用于将区块链网络中的交易数据打包为区块。

6.根据权利要求5所述的方法,其特征在于,所述根据所述第一测试结果的哈希值,确定区块生成节点,包括:

确定区块链网络中区块链节点的数量,并根据所述第一测试结果的哈希值对所述区块链节点的数量进行取模运算,得到目标数值;

获取所述区块链网络中多个区块链节点的节点标识,根据所述目标数值以及多个区块链节点的节点标识,从所述多个区块链节点中确定所述区块生成节点。

7.根据权利要求1所述的方法,所述获取区块生成节点随机算法程序,包括:

接收程序管理服务器发送的所述区块生成节点随机算法程序;

则所述方法还包括：

对所述区块生成节点随机算法程序进行验证，当判断所述区块生成节点随机算法程序的验证结果为验证通过时，执行所述区块生成节点随机算法程序的步骤。

8.一种程序测试装置，其特征在于，包括：

获取模块，用于获取区块生成节点随机算法程序，所述区块生成节点随机算法程序包括随机函数和随机种子；

运行模块，用于运行所述区块生成节点随机算法程序，得到第一测试结果；

处理模块，用于根据所述随机函数和所述随机种子生成测试程序，运行所述测试程序，得到第二测试结果；

确定模块，用于根据所述第一测试结果和所述第二测试结果确定所述区块生成节点随机算法程序的程序性能。

9.一种电子设备，其特征在于，包括存储器以及处理器，所述存储器存储一组程序代码，所述处理器调用所述存储器中存储的程序代码，用于执行1～7任一项操作。

10.一种计算机可读存储介质，其特征在于，所述计算机可读存储介质存储有计算机程序，所述计算机程序包括程序指令，所述程序指令当被处理器执行时使所述处理器执行如权利要求1～7任一项所述的方法。

一种程序测试方法、程序测试装置、电子设备及存储介质

技术领域

[0001] 本发明涉及互联网技术领域，尤其涉及一种程序测试方法、程序测试装置、电子设备及存储介质。

背景技术

[0002] 区块链是一种多方共同维护，使用密码学保证传输和访问安全，能够实现数据一致存储、难以篡改、防止抵赖的记账技术，也就是分布式账本技术。区块链技术中，是由区块生成节点(即记账节点)发起区块提案，其他节点才去验证区块，如果区块生成节点选举过于集中，会加大单点故障、关键节点作恶的中心化风险，因此区块生成节点的随机选择至关重要。

[0003] 目前，采用区块生成节点选择算法来确定区块生成节点，传统的针对区块生成节点选择算法随机性的测试方案是：运行区块生成节点选择算法程序，对程序输出结果运用频率检验、线性复杂度检验、近似熵检验、离散度计算等方法评定算法的随机性，但在运行区块生成节点选择算法之前无法预测程序输出结果，即现有方案无法解决区块生成节点选择算法的测试预测问题，进而会降低算法测试精度。因此，如何解决区块生成节点选择算法随机性的测试预测问题是当前亟待解决的问题。

发明内容

[0004] 本发明实施例提出了一种程序测试方法、程序测试装置、电子设备及存储介质，解决了区块生成节点选择算法随机性的测试预测问题，以准确验证程序测试节点选择算法的随机性和可靠性。

[0005] 本发明实施例提供了一种程序测试方法，该方法包括：

[0006] 获取区块生成节点随机算法程序，区块生成节点随机算法程序包括随机函数和随机种子；

[0007] 运行区块生成节点随机算法程序，得到第一测试结果；

[0008] 根据随机函数和随机种子生成测试程序，运行测试程序，得到第二测试结果；

[0009] 根据第一测试结果和第二测试结果确定区块生成节点随机算法程序的程序性能。

[0010] 本发明实施例提供了一种程序测试装置，该装置具有实现所述的数据处理方法的功能。所述功能可以通过硬件实现，也可以通过硬件执行相应的软件实现。所述硬件或软件包括：

[0011] 获取模块，用于获取区块生成节点随机算法程序，所述区块生成节点随机算法程序包括随机函数和随机种子；

[0012] 运行模块，用于运行所述区块生成节点随机算法程序，得到第一测试结果；

[0013] 处理模块，用于根据所述随机函数和所述随机种子生成测试程序，运行所述测试程序，得到第二测试结果；

[0014] 确定模块，用于根据所述第一测试结果和所述第二测试结果确定所述区块生成节

点随机算法程序的程序性能。

[0015] 本发明实施例提供了一种电子设备，该设备包括处理器、输入设备、输出设备和存储器，所述处理器、输入设备、输出设备和存储器相互连接，其中，所述存储器用于存储计算机程序，所述计算机程序包括程序指令，所述处理器被配置用于调用所述程序指令，用于执行上述程序测试方法所涉及到的操作。

[0016] 本发明实施例提供了一种计算机可读存储介质，用于储存为电子设备所用的计算机程序指令，其包含用于执行上述程序测试方法所涉及的程序。

[0017] 通过本发明实施例所提供的程序测试方法、程序测试装置、电子设备及存储介质，通过区块生成节点随机算法中的损失函数和随机种子生成测试程序，运行该测试程序，得到第二测试结果。第二测试结果为第一测试结果的预期输出结果，即在测试过程中有明确的预测输出结果，可解决区块生成节点选择算法随机性的测试预测问题，从而可以提高对区块生成节点随机算法的测试精度；再有，通过自动生成相应的测试程序，可以提高测试效率。

附图说明

[0018] 为了更清楚地说明本发明实施例技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图是本发明的一些实施例，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。

[0019] 图1A是本发明实施例提供的一种区块链数据共享系统的结构示意图；

[0020] 图1B是本发明实施例提供的一种区块链的结构示意图；

[0021] 图1C是本发明实施例提供的一种生成新区块的流程示意图；

[0022] 图2是本发明实施例提供的一种生成程序测试系统的架构示意图；

[0023] 图3是本发明实施例提供的一种程序测试方法的流程示意图；

[0024] 图4A是本发明实施例提供的一种随机函数的示意图；

[0025] 图4B是本发明实施例提供的一种随机种子的示意图；

[0026] 图5是本发明实施例提供的一种测试程序的示意图；

[0027] 图6A是本发明实施例提供的一种编写测试程序的模板文件的示意图；

[0028] 图6B是本发明实施例提供的另一种测试程序的示意图；

[0029] 图7为本发明实施例提供的一种测试报告的示意图；

[0030] 图8为本发明实施例提供的一种使用区块链生成节点随机算法的流程图；

[0031] 图9是本发明实施例提供的一种程序测试装置的结构示意图；

[0032] 图10是本发明实施例提供的一种电子设备的结构示意图。

具体实施方式

[0033] 下面将结合本发明实施例中的附图，对本发明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发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0034] 区块链(Block Chain)是一种多方共同维护，使用密码学保证传输和访问安全，能

够实现数据一致存储、难以篡改、防止抵赖的记账技术，也就是分布式账本技术。

[0035] 请参见图1A，图1A是本发明实施例提供的一种区块链数据共享系统的结构示意图，数据共享系统100是指用于进行节点与节点之间数据共享的系统，该数据共享系统中可以包括多个节点101，多个节点101可以是指数据共享系统中各个客户端。每个节点101在进行正常工作可以接收到输入信息，并基于接收到的输入信息维护该数据共享系统内的共享数据。为了保证数据共享系统内的信息互通，数据共享系统中的每个节点之间可以存在信息连接，节点之间可以通过上述信息连接进行信息传输。例如，当数据共享系统中的任意节点接收到输入信息时，数据共享系统中的其他节点便根据共识算法获取该输入信息，将该输入信息作为共享数据中的数据进行存储，使得数据共享系统中全部节点上存储的数据均一致。

[0036] 对于数据共享系统中的每个节点，均具有与其对应的节点标识，而且数据共享系统中的每个节点均可以存储有数据共享系统中其他节点的节点标识，以便后续根据其他节点的节点标识，将生成的区块广播至数据共享系统中的其他节点。每个节点中可维护一个如下表所示的节点标识列表，将节点名称和节点标识对应存储至该节点标识列表中。其中，节点标识可为IP(Internet Protocol，网络之间互联的协议)地址以及其他任一种能够用于标识该节点的信息，表1中仅以IP地址为例进行说明。

[0037]

节点名称	节点标识
节点1	117.114.151.174
节点2	117.116.189.145
…	…
节点N	119.123.789.258

[0038] 区块链数据共享系统中的每个节点均存储一条相同的区块链。区块链由多个区块组成，请参见图1B，图1B是本发明实施例提供的一种区块链的结构示意图，区块链由多个区块组成，创始块中包括区块头和区块主体，区块头中存储有输入信息特征值、版本号、时间戳和难度值，区块主体中存储有输入信息；创始块的下一区块以创始块为父区块，下一区块中同样包括区块头和区块主体，区块头中存储有当前区块的输入信息特征值、父区块的区块头特征值、版本号、时间戳和难度值，并以此类推，使得区块链中每个区块中存储的区块数据均与父区块中存储的区块数据存在关联，保证了区块中输入信息的安全性。

[0039] 在生成区块链中的各个区块时，请参见图1C，图1C是本发明实施例提供的一种生成新区块的流程示意图。区块链所在的节点在接收到输入信息时，对输入信息进行校验，完成校验后，将输入信息存储至内存池中，并更新其用于记录输入信息的哈希树；之后，将更新时间戳更新为接收到输入信息的时间，并尝试不同的随机数，多次进行特征值计算，使得计算得到的特征值可以满足下述公式(1)：

[0040] SHA256(SHA256(version+prev_hash+merkle_root+ntime+nbits+x)) (1)

[0041] 公式(1)中，安全散列算法256(Secure Hash Algorithm256，SHA256)为计算特征值所用的特征值算法；version(版本号)为区块链中相关区块协议的版本信息；prev_hash为当前区块的父区块的区块头特征值；merkle_root为输入信息的特征值；ntime为更新时间戳的更新时间；nbits为当前难度，在一段时间内为定值，并在超出固定时间段后再次进行确定；x为随机数；TARGET为特征值阈值，该特征值阈值可以根据nbits确定得到。

[0042] 这样，当计算得到满足上述公式的随机数时，便可将信息对应存储，生成区块头和区块主体，得到当前区块。随后，区块链所在节点根据区块链数据共享系统中其他节点的节点标识，将新生成的区块分别发送给其所在的区块链数据共享系统中的其他节点，由其他节点对新生成的区块进行校验，并在完成校验后将新生成的区块添加至其存储的区块链中。

[0043] 区块链的生命周期如下：1.客户端发送交易给节点，节点通过P2P网络广播交易到其他节点；2.共识机制随机选出区块生成节点；3.区块生成节点将若干交易列表打包组装成区块结构，并广播给其他节点；4.其他节点收到区块后，验证区块内的交易，并广播投票信息；5.针对该区块达成共识后，提交该区块到区块链存储层，开始下一轮选举和共识。为了达到多中心记账的目的，区块生成节点的选择需要充分的随机性和可验证。正是因为区块生成节点发起区块提案，其他节点才去验证区块，区块生成节点的随机选择至关重要，当节点选举过于集中，会加大单点故障、关键节点作恶的中心化风险，所以区块生成节点选择的随机性需要得到可靠性验证。传统的针对算法随机性的测试，往往集中在对算法跑出来的随机结果的检查，比如频率检验、游程检验、线性复杂度检验、离散度检验等，这些方法通常是从概率的角度来观察，解决不了Oracle(测试预测)问题。

[0044] 基于此，本发明实施例提出了一种程序测试方法、程序测试装置、电子设备及存储介质。其中，本申请的程序测试方法属于蜕变测试技术，蜕变测试技术是一种用来缓解“测试准则问题”的软件测试技术，测试准则是一种让测试人员判定程序是否能通过测试的机制，当测试人员对于所选择的测试用例难以确定预期的正确结果，或无法判定程序输出是否满足预期的结果时，便认为存在“测试准则问题”。该程序测试方法具体包括：获取区块生成节点随机算法程序，区块生成节点随机算法程序包括随机函数和随机种子；运行区块生成节点随机算法程序，得到第一测试结果；根据随机函数和随机种子生成测试程序，运行测试程序，得到第二测试结果；根据第一测试结果和所述第二测试结果确定区块生成节点随机算法程序的程序性能。

[0045] 通过本发明实施例所提供的用于区块链区块生成节点随机选择算法可验证的程序测试方法，可以解决传统的针对区块生成节点选举算法随机性的测试中存在的测试准则问题，在无法准确预测随机数的情况下，用数学分析+定理推导，对节点选择随机性的可靠性进行验证；并且，引入自动化框架，自动挖掘区块链项目程序中的随机函数和随机种子，自动生成相应的测试执行程序，运行完生成对应的可靠性测试报告。

[0046] 基于以上分析，请参见图2，图2是本发明实施例提供的一种生成程序测试系统的架构示意图。如图2所示，该系统架构图至少可以包括：程序管理服务器210，测试服务器220，至少一个服务器230，区块生成节点240，至少一个节点250。其中，程序管理服务器210与测试服务器220相连，程序管理服务器210与测试服务器220之间可进行通信，需要说明的是，测试服务器220可以是所有服务器中除程序管理服务器之外的任意服务器，例如测试服务器也可以为至少一个服务器230中的任意服务器，本发明对此不作限定。

[0047] 在一种实现方式中，程序管理服务器210将程序发送给距离程序管理服务器210最近的服务器，假设为服务器220，则服务器220即为测试服务器220。

[0048] 在一种实现方式中，程序管理服务器210将程序发送至与程序管理服务器210通信质量最好的服务器，假设为服务器220，则服务器220即为测试服务器220。

[0049] 在一种实现方式中，区块生成节点240与至少一个节点250处于区块链网络中，各个区块链节点上都存储有智能合约，客户端发送交易给节点，节点通过P2P网络广播交易到其他节点，其中，P2P网络即对等网络，该网络中的节点角色既可以是客户端，也可以是服务器，即该网络中的每一台计算机既能充当网络服务的请求者，又对其它计算机的请求做出响应，提供资源、服务和内容；共识机制随机选出区块生成节点，其中，共识机制中所使用到的共识算法包括但不限于工作量证明(Proof of Work，PoW)算法、权益证明(Proof of Stake，PoS)算法、授权权益证明(Delegated Proof of Stake，DPoS)算法、实用拜占庭容错(Practical Byzantine Fault Tolerance，PBFT)算法等，以区块生成节点为区块生成节点240为例，区块生成节点240将若干交易列表打包组装成区块结构，并广播给其他节点250；其他节点250收到区块后，验证区块内的交易，并广播投票信息；针对该区块达成共识后，提交该区块到区块链存储层，开始下一轮选举和共识。

[0050] 可以理解的是，本发明实施例描述的程序测试系统是为了更加清楚的说明本发明实施例的技术方案，并不构成对于本发明实施例提供的技术方案的限定，本领域普通技术人员可知，随着系统架构的演变和新业务场景的出现，本发明实施例提供的技术方案对于类似的技术问题，同样适用。

[0051] 请参见图3，图3是本发明实施例所提供的一种程序测试方法的流程示意图。该方法的操作步骤可由测试服务器执行，该方法包括但不限于如下步骤S310～S340：

[0052] 步骤S310：测试服务器获取区块生成节点随机算法程序，区块生成节点随机算法程序包括随机函数和随机种子。

[0053] 如图4A和4B，随机函数如图4A所示，随机种子如图4B所示。其中，程序管理服务器可以为REQ平台，即软件研发提测流程管理平台，该平台可用于管理软件版本开发、提测、缺陷提交、回归测试、发布整个生命周期等。

[0054] 在一种实现方式中，测试服务器接收程序管理服务器发送的区块生成节点随机算法程序请求时，请求中可以包含区块生成节点随机算法程序以及程序管理服务器的身份信息，测试服务器对该程序管理服务器的身份进行校验。例如，测试服务器获取程序管理服务器的身份信息可以是程序管理服务器的数字签名，其中，数字签名是指程序管理服务器根据程序管理服务器的私钥对区块生成节点随机算法程序进行签名处理得到的。测试服务器接收到区块生成节点随机算法程序请求之后，可以根据程序管理服务器的公钥，对区块生成节点随机算法程序请求中的数字签名进行校验，若校验成功，测试服务器对程序管理服务器的身份验证通过，测试服务器获取区块生成节点随机算法程序请求中的区块生成节点随机算法程序。通过这种方式，有利于提高区块生成节点随机算法程序的安全性。

[0055] 在一种实现方式中，测试服务器接收程序管理服务器发送的区块生成节点随机算法程序请求时，区块生成节点随机算法程序请求中可以包含区块生成节点随机算法程序以及程序管理服务器的服务器标识，测试服务器根据程序管理服务器的服务器标识对程序管理服务器的权限进行验证，若测试服务器对程序管理服务器的权限验证通过，测试服务器获取区块生成节点随机算法程序请求中的区块生成节点随机算法程序。通过这种方式，有利于提高区块生成节点随机算法程序的可信度。

[0056] 在一种实现方式中，测试服务器获取到区块生成节点随机算法程序之后，测试服务器对区块生成节点随机算法程序进行验证，当判断区块生成节点随机算法程序的验证结

果为验证通过时，运行区块生成节点随机算法程序。

[0057] 具体实现时，测试服务器将区块生成节点随机算法程序发送给网络中的各个服务器，各个服务器对该区块生成节点随机算法程序的进行校验，具体可以为校验该区块生成节点随机算法程序的安全性，以及格式、语法的正确性等基础性问题。测试服务器接收来自各个服务器发送的校验结果，测试服务器统计校验结果为校验成功的服务器的数量，测试服务器判断校验结果为校验成功的服务器的数量是否大于预设数量阈值，当校验成功的服务器的数量大于预设数量阈值时，确定对区块生成节点随机算法程序的校验通过，触发测试程序服务器运行区块生成节点随机算法程序的操作。

[0058] 需要说明的是，预设数量阈值可以有多种参数形式，如正整数、或者百分比，本发明对此不作限定。

[0059] 在一种实现方式中，若预设数量阈值为5时，当测试服务器判断出校验结果为校验成功的服务器的数量为6时，此时校验成功的服务器的数量大于预设数量阈值5，说明该区块生成节点随机算法程序可信度很高，确定校验通过；当测试服务器判断出校验结果为校验成功的服务器的数量为5，此时校验成功的服务器的数量等于预设数量阈值5，说明该教育信息可信度比较高，确定校验通过；当测试服务器判断出校验结果为校验成功的服务器的数量为3，此时校验成功的服务器的数量小于预设数量阈值5，说明该区块生成节点随机算法程序可信度不够高，测试服务器则删除该区块生成节点随机算法程序。

[0060] 步骤S320：测试服务器运行区块生成节点随机算法程序，得到第一测试结果。

[0061] 其中，该测试结果可以为随机数，也可以是随机数组或者随机序列等。假设第一测试结果为一行十列的随机数组，生成的该随机数组具体为[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]。

[0062] 步骤S330：测试服务器根据随机函数和随机种子生成测试程序，运行测试程序，得到第二测试结果。

[0063] 在一种实现方式中，测试服务器获取与区块生成节点随机算法程序对应的测试脚本，并将随机函数和随机种子添加至测试脚本；测试服务器对添加随机函数和随机种子后的测试脚本进行模板引擎渲染处理，得到测试程序。如图5所示，测试服务器根据获取到的随机函数和随机种子，采用和被测程序相同的随机函数和随机种子，自动生成测试程序。

[0064] 在一种实现方式中，本发明实施例采用python的模板库Jinja2来生成测试程序，其中，Jinja2是python web框架Flask作者开发的一个模板系统，起初是仿django模板的一个模板引擎，为Flask提供模板支持，Jinja2使用BSD授权。Jinja2是基于python的模板引擎，功能比较类似于于PHP的smarty，J2ee的Freemarker和velocity。它能完全支持unicode，并具有集成的沙箱执行环境，应用广泛。开发流程如下：

[0065] (1)下载安装Jinja2

[0066] pip3 install jinja2

[0067] (2)编写测试程序的模板文件，如图6A所示。

[0068] (3)经过jinja2模板引擎渲染后，生成测试程序template.render()，如图6B所示。

[0069] 需要说明的是，步骤S320和步骤S330的执行顺序没有限定。

[0070] 步骤S340：测试服务器根据第一测试结果和第二测试结果确定区块生成节点随机

算法程序的程序性能。

[0071] 在一种实现方式中，区块生成节点随机算法程序的程序性能包括随机准确程度。测试服务器统计第一测试结果和第二测试结果的结果匹配度。其中，结果匹配度的取值范围为(0,1]。

[0072] 举例来说，假设第一测试结果为：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]，第二测试结果为：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]。测试服务器将第一测试结果中的各个子随机数与第二测试结果中的各个子随机数进行对位依次比较，即测试服务器将第一测试结果中的第一位子随机数“4.17022005e-01”与第一测试结果中的第一位子随机数“4.17022005e-01”进行比较；测试服务器将第一测试结果中的第二位子随机数“7.20324493e-01”与第一测试结果中的第二位子随机数“7.20324493e-01”进行比较；以此类推，最终测试服务器将第一测试结果中的十个子随机数与第二测试结果中的十个子随机数依次比较，得到结果匹配度。需要说明的是，当满足第一测试结果中的各个子随机数分别与第二测试结果中对应的各个子随机数相等时，测试服务器确定第一测试结果和第二测试结果的结果匹配度为1。

[0073] 在一种实现方式中，第二测试结果包括N个单位测试结果，N是正整数，N个单位测试结果是运行N次所述测试程序后所输出的测试结果。测试程序服务器统计第一测试结果和N个单位测试结果的N个单位匹配度，测试服务器根据N个单位匹配度确定所述结果匹配度。

[0074] 举例来说，假设N＝10，第二测试结果包括的10个单位测试结果如表1所示。

[0075] 表1

[0076]

N	随机数组									
1	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
2	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
3	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
4	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
5	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
6	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
7	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
8	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
9	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01
10	4.17022005e-01	7.20324493e-01	1.14374817e-04	3.02332573e-01	1.46755891e-01	9.23385948e-02	1.86260211e-01	3.45560727e-01	3.96767474e-01	5.38816734e-01

[0077] 如表1所示，测试服务器统计第一测试结果[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]和10个单位测试结果的10个单位匹配度，按照上述方法所述，测试服务器将第一测试结果[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]与第一单位测试结果[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01 1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]中的各个子随机数依次对位进行比较，得到第一单位匹配度为1，其中，第一单位测试结果为第二测试结果中的第一轮测试结果。由于第二测试结果中的各个单位测试结果都相同，所以测试服务器判断10个单位匹配度均等于1，根据10个单位匹配度确定结果匹配度为1。

[0078] 在一种实现方式中，测试服务器根据结果匹配度确定随机准确程度，测试服务器将随机准确程度、第一测试结果以及第二测试结果组合为测试报告，输出测试报告。如图7所示，图7为本发明实施例所提供的一种测试报告的示意图。举例来说，若结果匹配度等于

1,则随机准确程度高,若结果匹配度小于1,则随机准确程度低。

[0079] 在一种实现方式中,当随机准确程度大于预设程度阈值时,测试服务器对第一测试结果进行哈希运算,得到第一测试结果的哈希值。其中,本发明实施例所指的预设程度阈值为1,即需要满足结果匹配度等于1的情况,才可以触发执行测试服务器对第一测试结果进行哈希运算。其中,哈希算法包括但不限于消息摘要算法第五版(Message-Digest Algorithm 5,MD5)、安全散列算法256(Secure Hash Algorithm256,SHA256)、安全散列算法512(Secure Hash Algorithm512,SHA512)、安全散列算法384(Secure Hash Algorithm384,SHA384)、安全散列算法1(Secure Hash Algorithm 1,SHA-1)等。测试服务器根据第一测试结果的哈希值,确定区块生成节点,区块生成节点用于将区块链上的区块生成节点随机算法程序打包为区块。

[0080] 在一种实现方式中,当随机准确程度大于预设程度阈值,测试服务器获取业务随机种子,测试服务器将区块生成节点随机算法程序中的随机种子替换为业务随机种子,将替换后的区块生成节点随机算法程序作为区块生成节点业务随机算法程序。测试服务器运行区块生成节点业务随机算法程序,得到随机结果。需要说明的是,随机结果可以为随机数、随机数组或者随机序列,本发明对此不作限定。

[0081] 举例来说,测试服务器获取到的业务随机种子为seed(0),区块生成节点业务随机算法程序为:

[0082]
```
import numpy as np
num=0
np.random.seed(0) #业务随机种子
while （num<5）:
    print(np.random.rand(1,5))
    num +=1
```

[0083] 基于上述可知,测试服务器运行该区块生成节点业务随机算法程序,举例来说,得到的随机结果为:[0.5 0.1 0.3 0.3 0.7]。

[0084] 在一种实现方式中,测试服务器确定区块链网络中区块链节点的数量,并根据第一测试结果的哈希值对区块链节点的数量进行取模运算,得到目标数值;测试服务器获取区块链网络中多个区块链节点的节点标识,根据目标数值以及多个区块链节点的节点标识,从所述多个区块链节点中确定所述区块生成节点。

[0085] 在一种实现方式中,当第一测试结果为随机数组,则测试服务器将该随机数组中的各个子随机数进行相加,得到目标随机数。然后,测试服务器对目标随机数进行哈希运算,得到目标随机数的哈希值。最后,测试服务器根据目标随机数的哈希值对区块链节点的数量进行取模运算,得到目标数值;测试服务器获取区块链网络中多个区块链节点的节点标识,根据目标数值以及多个区块链节点的节点标识,从所述多个区块链节点中确定所述区块生成节点。

[0086] 在一种实现方式中,当第一测试结果为随机数组,则测试服务器根据该随机数组中的各个子随机数进行算术平均运算得到平均值,将该平均值作为目标随机数。然后,测试

服务器对目标随机数进行哈希运算，得到目标随机数的哈希值。最后，测试服务器根据目标随机数的哈希值对区块链节点的数量进行取模运算，得到目标数值；测试服务器获取区块链网络中多个区块链节点的节点标识，根据目标数值以及多个区块链节点的节点标识，从所述多个区块链节点中确定所述区块生成节点。

[0087] 在一种实现方式中，当第一测试结果为随机数，则测试服务器对随机数进行哈希运算，得到随机数的哈希值。最后，测试服务器根据随机数的哈希值对区块链节点的数量进行取模运算，得到目标数值；测试服务器获取区块链网络中多个区块链节点的节点标识，根据目标数值以及多个区块链节点的节点标识，从所述多个区块链节点中确定所述区块生成节点。

[0088] 举例来说，如图8所示，图8为本发明实施例所提供的一种使用区块链生成节点随机算法的流程图。假设测试服务器确定区块链网络中区块链节点的数量为4，测试服务器对随机数Y进行哈希运算得到哈希值为hash(Y)，测试服务器根据hash(Y)对区块链节点的数量4进行取模运算，得到B，则B的取值可能为0，1，2，3。

[0089] 在一种实现方式中，测试服务器获取区块链网络中多个区块链节点的节点标识之前，将多个区块链节点分别进行编号，当区块链节点的数量为4，则可以对区块链节点1编号为B0，区块链节点2编号为B1，区块链节点3编号为B2，区块链节点4编号为B3。需要说明的是，编号的方式以及表现形式可以为任意形式，例如还可以为x，y，z，m等，本发明对此不作限定。测试服务器根据获取到的多个区块链节点的编号以及目标数值B，从多个区块链节点中确定区块生成节点。

[0090] 在一种实现方式中，测试服务器根据获取到的多个区块链节点的编号以及目标数值B，从多个区块链节点中确定区块生成节点的方式可以为映射的方式。例如，确定的目标数值B＝0，则映射到区块链节点的节点标识为B0；确定的目标数值B＝1，则映射到区块链节点的节点标识为B1；确定的目标数值B＝2，则映射到区块链节点的节点标识为B2；确定的目标数值B＝3，则映射到区块链节点的节点标识为B3。

[0091] 通过本发明实施例所提供的程序测试方法，采用蜕变测试思想，在测试阶段，运行区块链项目中的区块生成节点随机算法程序，跑出第一测试结果，在无法准确预测随机数的情况下，用数学分析+定理推导，分析挖掘程序中与之匹配的随机种子，引入自动化框架，自动挖掘区块链项目程序中的随机函数和随机种子，自动生成相应的测试程序，运行测试程序，跑出第二测试结果，通过比较第一测试结果和第二测试结果，从而蜕变测试节点选择算法的随机性和可靠性。

[0092] 请参见图9，图9是本发明实施例提供一种程序测试装置的结构示意图。该程序测试装置用于执行图3对应的方法实施例中测试服务器所执行的步骤，该程序测试装置可包括：

[0093] 获取模块910，用于获取区块生成节点随机算法程序，所述区块生成节点随机算法程序包括随机函数和随机种子；

[0094] 运行模块920，用于运行所述区块生成节点随机算法程序，得到第一测试结果；

[0095] 处理模块930，用于根据所述随机函数和所述随机种子生成测试程序，运行所述测试程序，得到第二测试结果；

[0096] 确定模块940，用于根据所述第一测试结果和所述第二测试结果确定所述区块生

成节点随机算法程序的程序性能。

[0097] 在一种实现方式中，获取模块910、处理模块930根据随机函数和随机种子生成测试程序，包括：

[0098] 获取模块910获取与区块生成节点随机算法程序对应的测试脚本；

[0099] 处理模块930将随机函数和随机种子添加至测试脚本；

[0100] 处理模块930对添加随机函数和随机种子后的测试脚本进行模板引擎渲染处理，得到测试程序。

[0101] 在一种实现方式中，该程序测试装置还包括：统计模块950。

[0102] 统计模块950，用于统计第一测试结果和第二测试结果的结果匹配度。

[0103] 在一种实现方式中，程序性能包括随机准确程度；

[0104] 处理模块930、确定模块940、统计模块950根据第一测试结果和第二测试结果确定区块生成节点随机算法程序的程序性能，包括：

[0105] 统计模块950统计第一测试结果和第二测试结果的结果匹配度。

[0106] 确定模块940根据结果匹配度确定随机准确程度；

[0107] 处理模块930将随机准确程度、第一测试结果以及第二测试结果组合为测试报告，输出测试报告。

[0108] 在一种实现方式中，第二测试结果包括N个单位测试结果，N是正整数，N个单位测试结果是运行N次测试程序后所输出的测试结果；

[0109] 统计模块950统计第一测试结果和第二测试结果的结果匹配度，包括：

[0110] 统计模块950统计第一测试结果和N个单位测试结果的N个单位匹配度；

[0111] 确定模块940根据N个单位匹配度确定结果匹配度。

[0112] 在一种实现方式中，该程序测试装置还包括：计算模块960。

[0113] 计算模块960，用于当随机准确程度大于预设程度阈值时，对第一测试结果进行哈希运算，得到第一测试结果的哈希值；

[0114] 确定模块940根据第一测试结果的哈希值，确定区块生成节点；区块生成节点用于将区块链网络中的交易数据打包为区块。

[0115] 在一种实现方式中，确定模块940根据第一测试结果的哈希值，确定区块生成节点，包括：

[0116] 确定模块940确定区块链网络中区块链节点的数量，计算模块960根据第一测试结果的哈希值对区块链节点的数量进行取模运算，得到目标数值；

[0117] 获取模块910获取区块链网络中多个区块链节点的节点标识，确定模块940根据目标数值以及多个区块链节点的节点标识，从多个区块链节点中确定区块生成节点。

[0118] 在一种实现方式中，该程序测试装置还包括：接收模块970、验证模块980。

[0119] 接收模块970，用于接收程序管理服务器发送的所述区块生成节点随机算法程序；

[0120] 验证模块980，用于对区块生成节点随机算法程序进行验证；

[0121] 当判断区块生成节点随机算法程序的验证结果为验证通过时，处理模块930执行区块生成节点随机算法程序的步骤。

[0122] 请参见图10，图10是本发明实施例提供的一种电子设备的结构示意图，该电子设备用于执行图3对应的方法实施例中测试服务器所执行的步骤，该电子设备包括：一个或多

个处理器1010;一个或多个输入设备1020,一个或多个输出设备1030和存储器1040。上述处理器1010、输入设备1020、输出设备1030和存储器1040通过总线1050连接。存储器1020用于存储计算机程序,所述计算机程序包括程序指令,处理器1010、输入设备1020用于执行存储器1040存储的程序指令,执行以下操作:

[0123] 输入设备1020获取区块生成节点随机算法程序,区块生成节点随机算法程序包括随机函数和随机种子;

[0124] 处理器1010运行区块生成节点随机算法程序,得到第一测试结果;

[0125] 处理器1010根据随机函数和随机种子生成测试程序,运行测试程序,得到第二测试结果;

[0126] 处理器1010根据第一测试结果和第二测试结果确定区块生成节点随机算法程序的程序性能。

[0127] 在一种实现方式中,处理器1010、输入设备1020根据随机函数和随机种子生成测试程序,包括:

[0128] 输入设备1020获取与区块生成节点随机算法程序对应的测试脚本;

[0129] 处理器1010将随机函数和随机种子添加至测试脚本;

[0130] 处理器1010对添加随机函数和随机种子后的测试脚本进行模板引擎渲染处理,得到测试程序。

[0131] 在一种实现方式中,程序性能包括随机准确程度;

[0132] 处理器1010根据第一测试结果和第二测试结果确定区块生成节点随机算法程序的程序性能,包括:

[0133] 统计第一测试结果和第二测试结果的结果匹配度;

[0134] 根据结果匹配度确定随机准确程度;

[0135] 将随机准确程度、第一测试结果以及第二测试结果组合为测试报告,输出测试报告。

[0136] 在一种实现方式中,第二测试结果包括N个单位测试结果,N是正整数,N个单位测试结果是运行N次测试程序后所输出的测试结果;

[0137] 处理器1010统计第一测试结果和第二测试结果的结果匹配度,包括:

[0138] 统计第一测试结果和N个单位测试结果的N个单位匹配度;

[0139] 根据N个单位匹配度确定结果匹配度。

[0140] 在一种实现方式中,处理器1010统计第一测试结果和第二测试结果的结果匹配度,还包括:

[0141] 当随机准确程度大于预设程度阈值时,对第一测试结果进行哈希运算,得到第一测试结果的哈希值;

[0142] 根据第一测试结果的哈希值,确定区块生成节点;区块生成节点用于将区块链网络中的交易数据打包为区块。

[0143] 在一种实现方式中,处理器1010、输入设备1020根据第一测试结果的哈希值,确定区块生成节点,包括:

[0144] 确定区块链网络中区块链节点的数量,并根据第一测试结果的哈希值对区块链节点的数量进行取模运算,得到目标数值;

[0145] 输入设备1020获取区块链网络中多个区块链节点的节点标识，处理器1010根据目标数值以及多个区块链节点的节点标识，从多个区块链节点中确定区块生成节点。

[0146] 在一种实现方式中，处理器1010、输入设备1020获取区块生成节点随机算法程序，包括：

[0147] 输入设备1020接收程序管理服务器发送的区块生成节点随机算法程序；

[0148] 处理器1010对区块生成节点随机算法程序进行验证，当判断区块生成节点随机算法程序的验证结果为验证通过时，执行区块生成节点随机算法程序的步骤。

[0149] 本发明实施例中还提供一种计算机可读存储介质，所述计算机可读存储介质存储有计算机程序，所述计算机程序包括程序指令，所述程序指令被处理器执行时，可执行上述实施例中所执行的步骤。

[0150] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机程序来指令相关的硬件来完成，所述的程序可存储于一计算机可读取存储介质中，该程序在执行时，可包括如上述数据处理方法的实施例的流程。其中，所述的存储介质可为磁碟、光盘、只读存储记忆体(Read-Only Memory，ROM)或随机存储记忆体(Random Access Memory，RAM)等。

[0151] 以上所揭露的仅为本发明的部分实施例而已，当然不能以此来限定本发明之权利范围，本领域普通技术人员可以理解实现上述实施例的全部或部分流程，并依本发明权利要求所作的等同变化，仍属于发明所涵盖的范围。

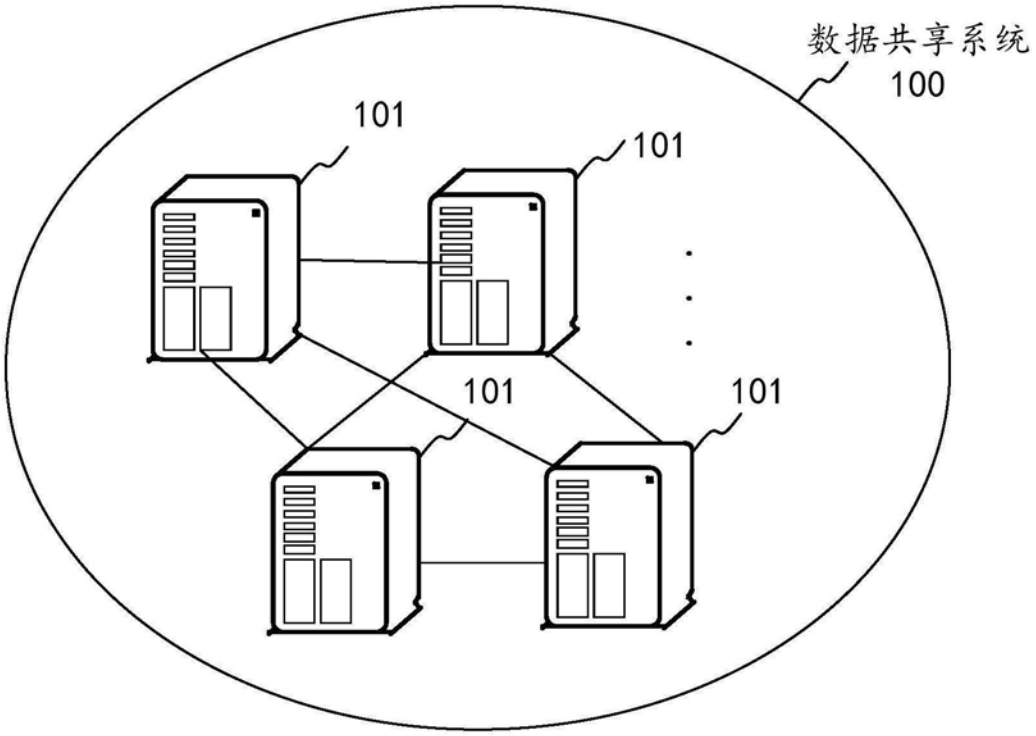

数据共享系统
100
101
101
101
101

图1A

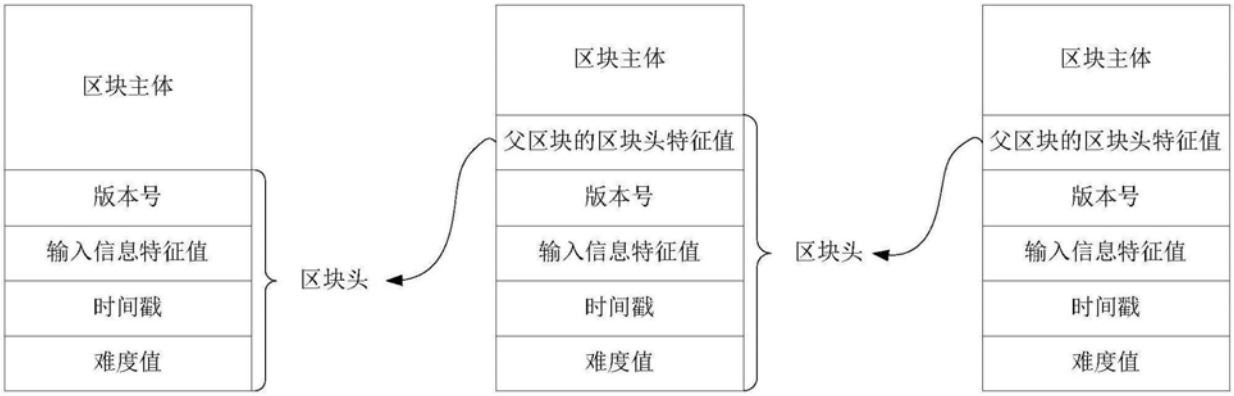

区块主体
版本号
输入信息特征值
时间戳
难度值
区块头
区块主体
父区块的区块头特征值
版本号
输入信息特征值
时间戳
难度值
区块头
区块主体
父区块的区块头特征值
版本号
输入信息特征值
时间戳
难度值

图1B

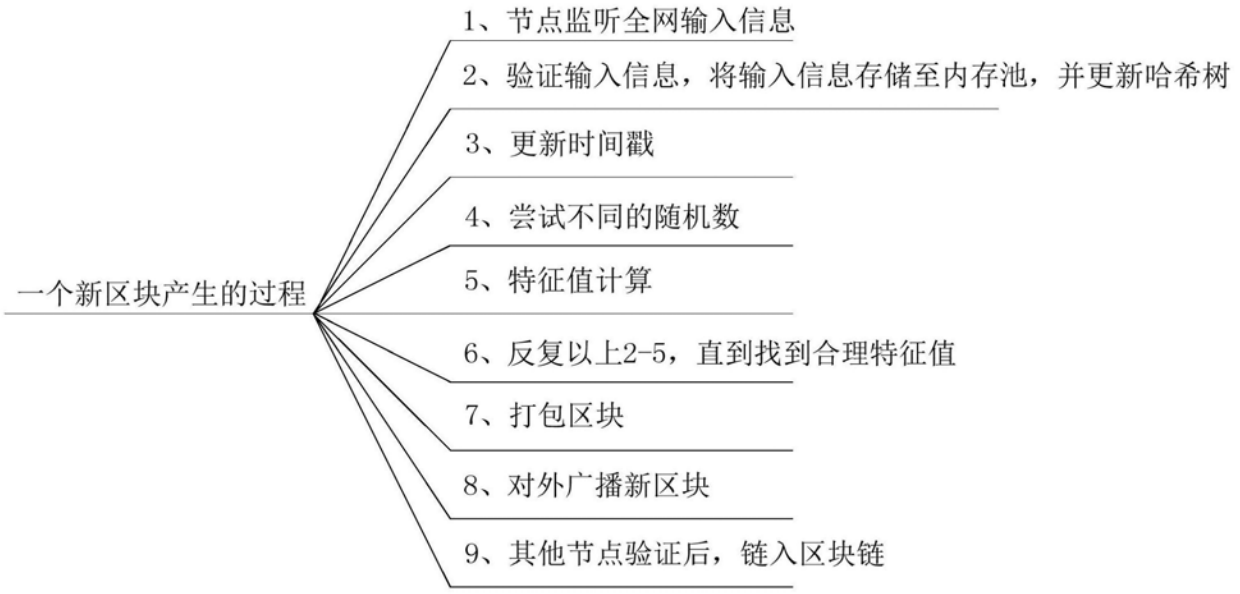

一个新区块产生的过程
1、节点监听全网输入信息
2、验证输入信息，将输入信息存储至内存池，并更新哈希树
3、更新时间戳
4、尝试不同的随机数
5、特征值计算
6、反复以上2-5，直到找到合理特征值
7、打包区块
8、对外广播新区块
9、其他节点验证后，链入区块链

图1C

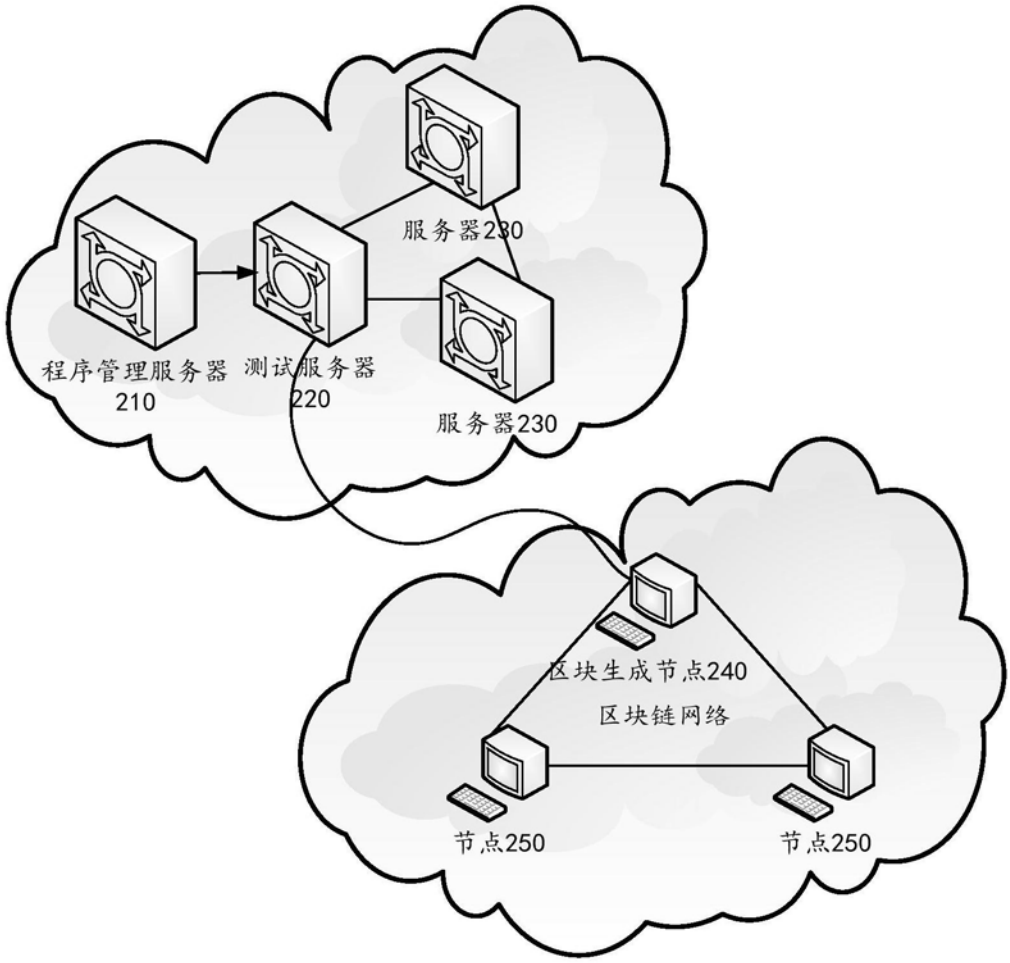

服务器230
程序管理服务器
210
测试服务器
220
服务器230
区块生成节点240
区块链网络
节点250
节点250

图2

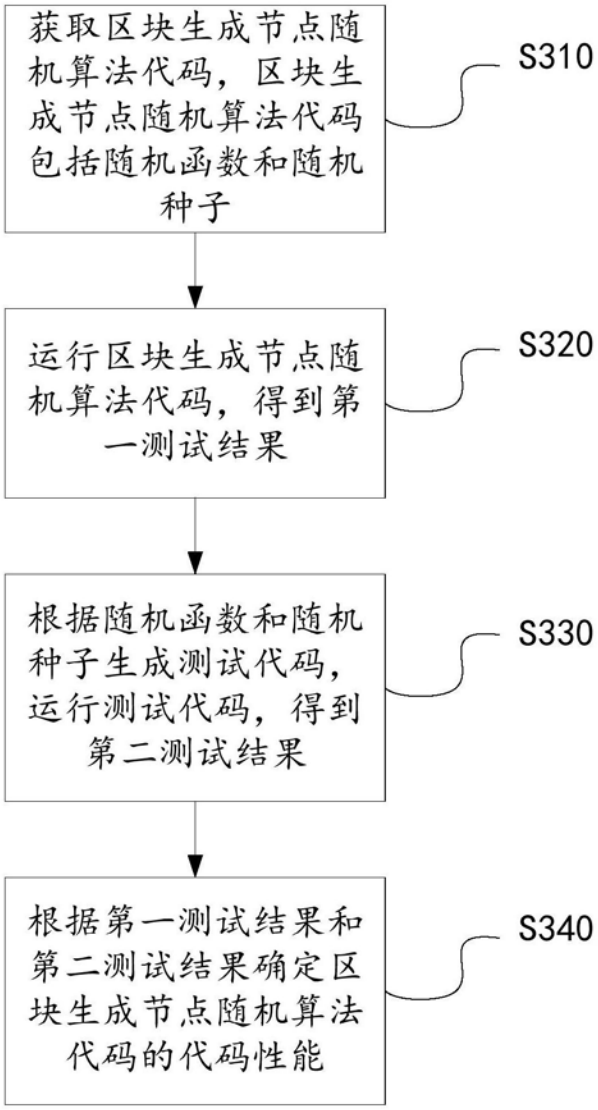


图3

```
import numpy as np
num=0
np.random.seed(1)
while（num<10）：
    print(np.random.rand(1,10))
    num +=1
```

随机函数

图4A

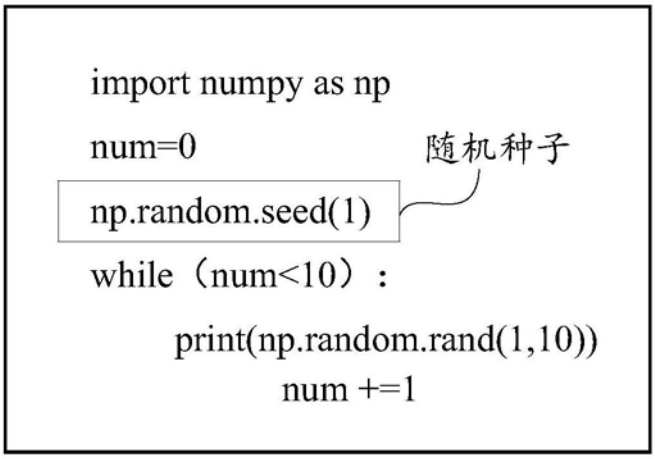


图4B

```
import numpy as np
num=0
while (num<10) :
np.random.seed(1)
    print(np.random.rand(1,10)) #得到一个范围从0到1的 1行10列的随机数
num +=1
```

图5

```
num=0
print(“----------测试报告----------”)
print(“测试函数：”+“{{rand_function}}”)
Print(“测试随机种子：”+“{{rand_seed}}”)
while (num<10) :
{{rand_seed}}
    print(“测试round：”+str(num+1)+“,”+“随机数：”+{{rand_function}}))
    num +=1
```

图6A

```
num=0
print("----------测试报告----------")
print("测试函数：" + "np.random.rand(1,10)")
print("测试随机种子：" + "np.random.seed(1)")
while (num<10) :
np.random.seed(1)
    print("测试round：" + str(num+1) + "," + "随机数：" + str(np.random.rand(1,10)))
    num +=1
```

图6B

```
----------测试报告----------
测试函数：np.random.rand(1,10)
测试随机种子：seed(1)
测试结果：pass
测试 round：1，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：2，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：3，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：4，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：5，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：6，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：7，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：8，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：9，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
测试 round：10，随机数：[4.17022005e-01 7.20324493e-01 1.14374817e-04 3.02332573e-01
1.46755891e-01 9.23385948e-02 1.86260211e-01 3.45560727e-01 3.96767474e-01 5.38816734e-01]
----------
```

图7

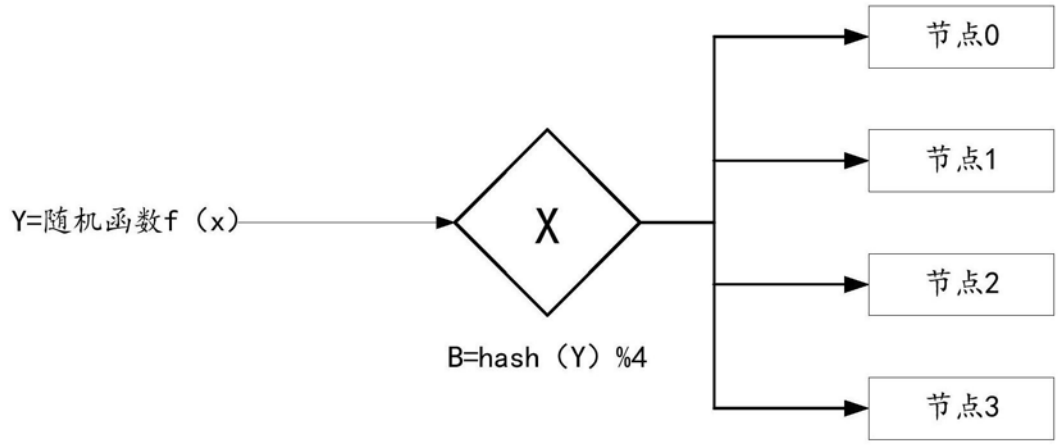

节点0
节点1
节点2
节点3
Y=随机函数f（x）
X
B=hash（Y）%4

图8

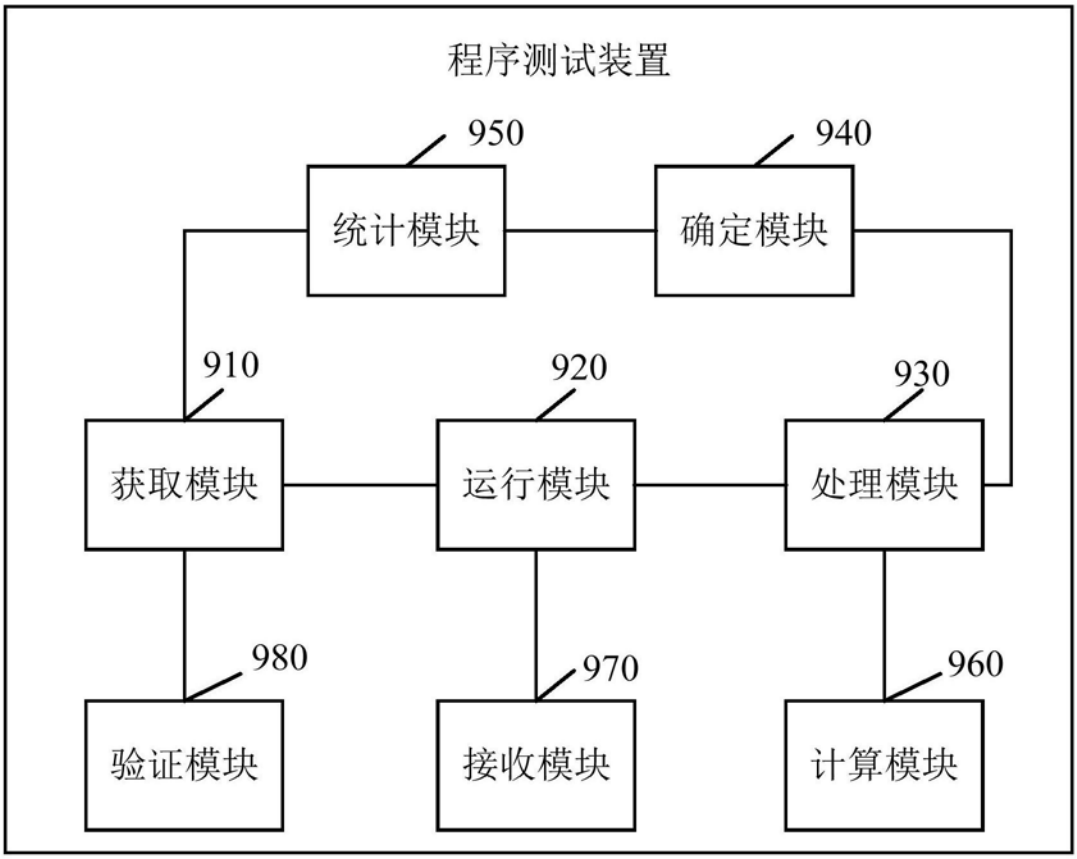

程序测试装置
950
统计模块
940
确定模块
910
获取模块
920
运行模块
930
处理模块
980
验证模块
970
接收模块
960
计算模块

图9

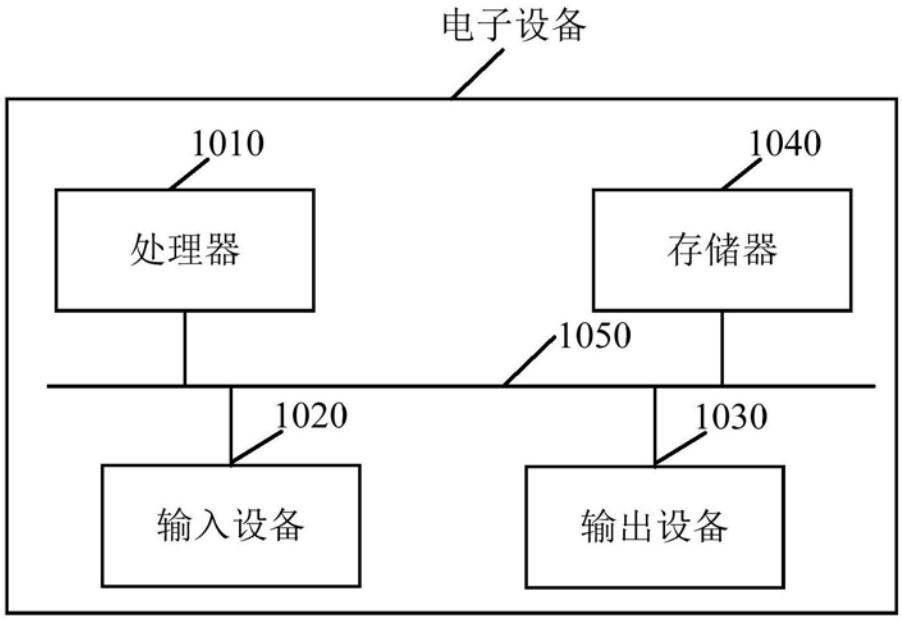


图10