华南理工大学

课程报告

课程名称：	企业软件项目实训
学生姓名：	秦华
学生学号：	201630665489
学生专业：	软件工程
开课学期：	2018-2019 第二学期

软件学院

2019 年 6 月

1. 区块链技术原理

1.1 总体概述

比特币是一种数字货币，以区块链为底层技术，基于区块链技术的应用不仅只有数字货币，还有智能合约、证券交易等等。本章以比特币为例，通过分析比特币的交易存储的全过程来介绍区块链的技术原理。

比特币作为一种开放的分布式密码货币系统，任何人都可以随时加入或离开，成为其中的一个节点，一般来说，每个节点都拥有一份完整的账本，即包含所有交易信息的一份完整的区块链。

比特币系统中的节点可以发送货币，也可以接收货币，还可以担任矿工节点 B 造区块，也可以担任验证节点 V 验证区块是否有效。以下提到的 Bi,Bj,Vi 等均为矿工节点 B、验证节点 V 对应的任一节点。

假设 Alice 向 Bob 发送了一定数量的比特币，这称为一笔交易。比特币系统中的运行全过程如下：

（1）Alice 与 Bob 的交易发生（具体过程参见 1.2 节）。

（2）交易广播到网络 。

（3）网络中的任意矿工节点 Bi 监听到 Alice 与 Bob 的交易。

（4）任意矿工 Bi 验证该笔交易是否合法（具体过程参见 1.3 节）。

（5）如果合法，任意矿工 Bi 将该交易放进自己的交易池中，并向自己周围的节点广播该笔交易；不合法则丢弃。

（6）任意矿工 Bi 从自己的交易池中取出一些交易组装成一个拟加入链条中的区块（“具体过程参见第 1.4 节”）。

（7）所有的矿工节点争夺记账权，设 Bj 争夺成功（“具体过程参见第 1.5 节”）。

（8）得到记账权的节点 Bj 将该区块加入链条中，并且向验证节点 V 广播它生成的这个区块。

（9）所有的 V 验证这个区块是否合法（“具体过程参见第 1.6 节“）。

（10）若合法接受则接受加入该新区块的区块链，并向其他节点广播，不合法则还是接受没有新区块的老区块链。

（11）若该区块被验证节点 V 所接受，Bj 获得奖励（“具体过程见第 1.7 节“）。

1.2 交易的产生

在比特币中，交易的内容都是比特币的转账。

其实交易也是一种数据结构，它包含着具体的数据，在比特币中，这些数据就是转账的金额，接收人的公钥，发起人的签名等。

每一个交易都要有另一个或多个交易作为来源。

交易的发起人是指创建这笔交易的比特币网络节点，具体来说就是利用输出到自己公钥上的交易与其他信息，通过执行一系列计算，得到一个新交易的节点；

交易的接收方就是这笔新交易所输出到的公钥地址。

现假设有这么一种情况，现在有用户 Alice 准备用创建一个交易 T，这个交易的来源是之前的交易T'，并且这笔交易输出给用户 Bob。

为了论述的方便，现在我们先来定义一些记号：

- T 是用户 Alice 创建的新交易, T'是 T 的源交易；
- h'是源交易T'的哈希值；
- (S_A,V_A)是网络中与 Alice 对应的公私钥，V_B是网络中与 Bob 对应的公钥；
- V 是交易 T 的具体转账金额；

Alice 的执行如下计算：

1. $\bar{h} \leftarrow \text{hash}(T',V_B,V)$

Alice 将源交易T'，Bob 的公钥和这次的金额 V 进行哈希运算得到交易信息摘要$\bar{h}$。

2. $\sigma \leftarrow SIG(S_A,\bar{h})$

Alice 将自己的私钥和上一步得到的摘要进行数字签名，得到签名 σ 。

3. $T \leftarrow (V_B,\text{h}',V,\sigma)$

Alice 将 Bob 的公钥，源交易的哈希值，本次交易的金额，和对新交易的签名组成新交易 T。

现在 Alice 可以将交易 T 向网络的其他用户广播了。

1.3 交易的验证

在比特币中，网络上某一节点 Bi 接收到了 Alice 广播的交易，然后开始验证这笔交易是否合法。

我们接着上一节的假设情况，Alice 广播
了他刚刚创建的交易 T，网络上另一用户 Bi
接收到了这个事务。

Bi 执行如下计算：

(1) $\text{bool} \leftarrow \text{Verify}(V_A,\text{V},\sigma\ ,\ T')$

首先将 Alice 的公钥和源交易T'中接收人的公钥对比，确认源交易确是输出给 Alice。再验证公钥，签名和金额等数据确保签名的真实性。只要确认了真实性，这笔交易就被认可为合法的。

1.4 区块链的生成

当交易发生之后，任一矿工节点 Bi 接收到新的交易 T 之后，验证交易合法后将交易放入交易池，交易池中的交易足够多的时候，开始造区块。接下来将详

细描述矿工节点 Bi 制造区块，形成区块链的过程。

1.4.1 结构说明

（1）区块链的结构

区块链是由区块连接而成的数据结构，而区块就是存储数据的逻辑块，存储着系统中的交易信息。简单来说，区块链是由一个一个的区块连接而成的，如图所示：

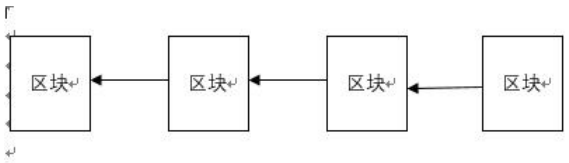

图 1 区块链结构简单示意图

那么区块具体是如何连接起来的呢？如图所示，每一个区块都包含着一个哈希指针，实际上就是上一区块的哈希值，该指针可以告诉我们上一个区块在哪里。

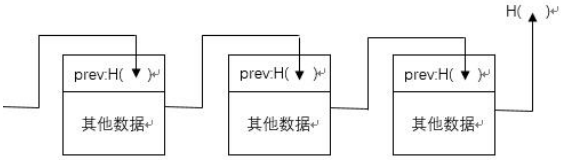

图 2 区块链连接结构示意图

（2）区块的结构

区块由区块头以及区块体组成。区块头由三组元数据组成，第一组是引用父区块哈希值的数据，即前面说到的哈希指针；第二组是难度目标、时间戳和 nounce,这三个与挖矿竞争相关；第三组是 merkle 树根，merkle 树是一种用来有效地总结区块中所有交易的数据结构。区块体则包含交易。[11]对于比特币来说，区块头还包含版本字段，主要来说明版本号，用于跟踪软件/协议的更新，由于版本字段跟本文叙述内容无关，所以这里忽略版本字段。

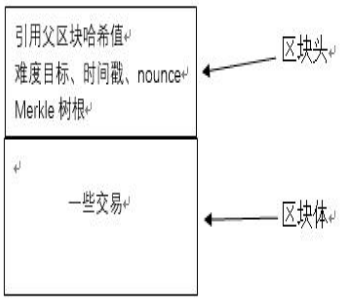

图 3 区块结构示意图

1.4.2 获取梅克尔根过程

前面提到区块的结构时，区块头里面有一个梅克尔根，那么梅克尔根是怎么来的呢？如图 3-9，对于区块体中所包含的每个交易都用哈希函数算出它们的哈希值，接着将每两个哈希值再做哈希运算，得到另外一批哈希值，再接着对每两个哈希值进行哈希运算，一直重复下去，直到最后得到唯一一个哈希值，该哈希值就是梅克尔根。

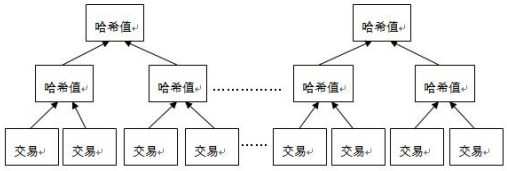

图 4 梅克尔树生成示意图

我们知道，网络中的每一个节点都可以获取一份完整的区块链，但是随着网络的运行，交易越来越多，区块链越来越长，如果将整份区块链都存进磁盘里，也需要耗费许多空间。事实上，比特币网络中有两种节点，分别是完全有效节点以及轻量级节点，完全有效节点存储了完整的区块链，而轻量级节点只存储区块头以及与自己相关的部分交易，由于区块头中有 Merkle 根，所以轻量级节点也可以验证区块的有效性，但轻量级节点所需要的空间却大大减小了[12]。

1.4.3 区块链形成过程

综上，区块链的总体结构如图所示：

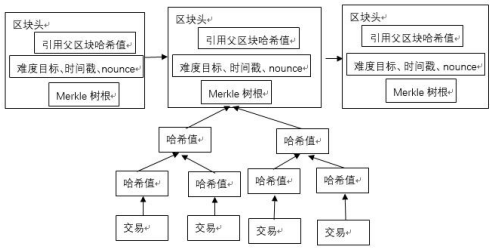

图 5 区块链总体结构示意图

区块链形成的具体过程如下：

（1）矿工节点 Bi 从交易池拿出一些交易，这些交易将要打包进候选区块。

（2）拿到前一区块的哈希值，向区块头添加前一区块哈希值。

（3）将交易经过层层哈希运算生成梅克尔树。

梅克尔树的生成过程见 1.4.2 节

（4）拿出梅克尔树的梅克尔根，向区块头加一个梅克尔根。

（5）时间戳服务器向区块头加时间戳。

时间戳的存在能够证实特定数据必然于某特定时刻是的确存在的[13]。但是区块链上判断某一数据是在另一个数据之前或之后写入记录的，这个次序是由哈希函

数指针决定的，并不由时间戳决定，因为时间戳是可以造假的或者不一定准确[12]。

（6）向区块头填充目标值以及难度值。

（7）向区块头填充 nounce，初始化为 0。

目标难度值与 nounce 与后面的共识机制有关，寻求 nounce 值的过程称为挖矿。尽管每个节点都可以成为矿工节点 B 参与挖矿，但是挖矿是一种竞争，并不是每一个节点最后都能挖矿成功。

1.5 共识机制

Alice 向 Bob 发送了比特币，这时这个交易要有效，就必须被存储到区块链这一本公共账本上。区块链是分布式的、全网公开的，位于网络中所有节点都可以决定将哪个交易记入这本账本。这时要给哪个节点记账权呢，才能使所有节点都承认这本账本记录的内容，从而达到同步。

共识机制可以解决哪个节点应该拥有记账权的问题。共识机制，指的是节点与节点之间在相互零信任的前提下，如何对一件事情达成一致，达到同步的状态。常用的共识机制有工作量证明、权益证明等等。其中比特币系统使用的共识机制是工作量证明。

工作量证明（Proof of Work，PoW）根据节点计算能力来选取节点。每一个节点都可以求解一个哈希函数谜题，由于哈希函数是谜题友好的，因此要找到符合条件的哈希值没有简单的方法，找到一个合适的哈希值唯一已知的方法是不停地随机试探直到搜索到一个有效的数，只要计算能力越强就越可能算出这个谜题，拿到记账权。因此算力更强大的节点更容易找到符合条件的哈希值。

1.4 节中 Bi 已经组装好了一个候选区块，这时这个区块要得到其他节点的认可，Bi 就要争取记账权。Bi 争取记账权的过程也就是要算出一个 nounce，也就是挖矿的过程。

随机数 nounce 是 32 位的，所以 Bi 可以尝试所有 32 位可能的取值。Bi 将 nounce、前一区块哈希值还有要添加这个区块的交易列表连接起来，组成一串字符串，然后用哈希函数计算该字符串的输出值。如果输出值小于目标值，那么说明成功找到了 nounce。

工作量证明可以使分布式节点达成共识，但也存在着缺陷。首先，工作量证明需要矿工节点耗费 CPU 算力来求解哈希谜题，造成了严重的资源耗费。另外，工作量证明的效率不高，一笔交易从发生到被完全确认，需要很长一段时间，大概一个小时左右。

1.6 区块的验证

假设在上一节的挖矿中，最终矿工节点 Bj 挖矿成功，找到了符合的 nounce

值。Bj 就可以向其他验证节点 V 广播它的区块，V 节点验证区块是否有效，V 检测的是

（1）区块的数据结构语法上有效

（2）区块头的哈希值小于目标难度

（3）区块时间戳早于验证时刻未来两个小时

（4）区块大小在长度限制之内

（5）第一个交易是 coinbase 交易

Coinbase 交易就是币基交易，矿工打包交易时打包的第一笔交易是向自己的地址转一定数量的比特币，这些比特币作为自己挖矿的奖励（详情可查看 1.7 节）

（6）验证区块内的每一个交易并确保它们的有效性（详情可查看 1.3 节）

若该区块通过验证，V 节点将它纳入区块链中，并向其它节点传播；没有通过验证则将其丢弃。[11]

1.7 激励机制

在之前的论述中，了解了交易的流程和区块链生成的过程。接下来我们对区块链的激励机制进行分析。比特币之所以是去中心化的，除了一些技术手段之外，还需要一些激励机制来实现。激励设计是一种将矿工的行动和网络安全保持一致的激励计划，同时还实现了货币的发行。

比特币中有两种激励机制，一种是区块奖励，而另一种称为交易费。

1.7.1 区块奖励

创建区块的那个节点可以在这个区块中加入一笔造币交易，即产生出一枚该创建者拥有的新电子货币。只有当造币交易纳入长期共识链时，奖励才会实现，才会被其他节点接受。这种激励机制使得激励节点想尽办法让其他节点延伸他们自己的区块，因此这样使得大部分节点都遵循了这种延长最长链的规则，从而激励所有节点都去遵循这个规则。

区块生成以十分钟为一个间隔，也就是十分钟生成一个区块。区块奖励的数量每四年减半一次。[1]开始时每个区块奖励 50 个比特币，然后减半为每个区块奖励 25 个比特币，随后是 12.5 个比特币……根据区块的生成速度，生成 210000 比特币大约需要四年。当年比特币总量为，此年之前的比特币总量加上当年生成的比特币数量（当年生成区块的数量与区块奖励之积）。

基于这种算法，比特币挖矿奖励以指数方式递减，直到 2140 年。届时所有的比特币全部发行完毕。换句话说在 2140 年之后，不会再有新的比特币产生。到那个时候，新的区块不在包含比特币奖励，矿工的收益全部来自于交易费，交易费将再下一小节加以论述。

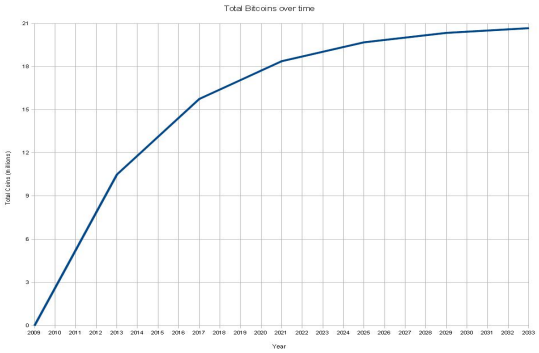

图 6 比特币的总供应量

1.7.2 交易费

在矿工把交易打包进是需要付出代价的，交易费就是用来补偿矿工处理交易时所付出的代价。

由此交易费设定为第一个创建区块把交易放进区块链的人可以取得的交易输入值和输出值的差额。首先，比较输入值和输出值的大小，当输入值大于输出值时，才进行交易 。当满足条件时，计算出交易费（即输入值减去输出值）。最后，交易费由交易发出方交给打包这笔交易到区块的矿工，也就是说，这笔交易费将会增加到区块的奖励中。

交易费的金额可以自己设定，也可以不设定，也存在默认的交易费政策。但是如果支付的交易费更高，奖励越多，这就意味着交易将更快、更可靠地传播和记录。目前，交易费占矿工收入不足 1%，因此，大部分收益仍来自挖矿所得的比特币奖励，大部分矿工也遵循默认的交易费政策。然而随着挖矿奖励的递减，交易费在矿工收益中所占的比重将会逐渐增加。在未来，所有的矿工收益都将由交易费构成。

2. 联盟链和公有链的异同

公有链向所有人开放，联盟链向授权的组织或机构开放。

公有链，也就是公共区块链(Public blockchains)，是指全世界任何一个人都可以读取、任何一个人都可以发送交易且交易能够获得有效确认的共识区块链。公有链上的各个节点可以自由加入和退出网络，并参加链上数据的读写，读写时以扁平的拓扑结构互联互通，网络中不存在任何中心化的服务端节点。

联盟链，即联盟区块链（Consortium blockchains），是指有若干组织或机构共同参与管理的区块链，每个组织或机构控制一个或多个节点，共同记录交易数据，并且只有这些组织和机构能够对联盟链中的数据进行读写和发送交易。联盟链的各个节点通常有与之对应的实体机构组织，通过授权后才能加入与退出网络。各机构组织组成利益相关的联盟，共同维护区块链的健康运转。

3. Gas 在智能合约中的作用

“gas”是以太坊使用的特殊单位的名称。它衡量一个动作或一系列动作需要执行多少“工作”：例如，计算一个 Keccak256 密码散列，每计算一次散列需

要 30 个气体，每 256 位 数据被哈希。 Ethereum 平台上的一项交易或合同可以执行的每项操作都会花费一定数量的天然气，其运营所需的计算资源比计算资源要求较少的运算需要更多的天然气。

gas 的重要性在于它有助于确保提交给网络的交易支付适当的费用。 通过要求交易支付每个操作的执行（或导致合同执行），我们确保网络不会因为执行大量对任何人无价值的密集工作而陷入困境。 这与比特币交易费用不同，它仅基于交易的千字节大小。 由于以太坊允许运行任意复杂的计算机代码，所以短的代码实际上可能导致大量计算工作的完成。 所以衡量直接完成的工作非常重要，而不是仅仅根据交易或合同的长度选择费用。

所以，如果 gas 基本上是交易费用，那么你如何支付？ 这是一个棘手的地方。 虽然 gas 是一个可以测量物质的单位，但 gas 并没有任何实际的标志。 也就是说，你不能拥有 1000gas。 相反，gas 只存在于以太坊虚拟机内部，作为正在执行多少工作的计数。 在实际支付 gas 时，交易费用是 ether 的一定数量，以太坊网络上的内置令牌和矿工奖励生产块的令牌。

起初这可能看起来很奇怪。 为什么不直接用 ether 衡量成本？ 答案是，就像比特币一样，以太网的市场价格可能会迅速变化！ 但是计算的代价并不是因为以太的价格变化而上升或下降的。 所以将计算价格与以太币的价格区分开来是很有用的，这样每次市场走势就不需要改变操作成本。

EVM 中的操作具有 gas 成本，但 gas 本身也具有以 ether 的 gas 价格。 每笔交易都规定了每个 gas 单位愿意支付的 gas 价格，从而使市场能够决定 gas 价格和计算成本（以天然气计量）之间的关系。 这是两者的总和，即所用 gas 总量乘以 gas price，得到交易支付的全部费用。

如果我在我的交易中设定的 gas price 太低，那么没有人会在第一时间去管理我的交易。 它不会被矿工包括在区块链中。但如果我提供一个可以接受的天然气价格，那么我的交易就会产生如此多的计算工作，以至于合并后的天然气成本超过了我所附加的费用数额，那么这个天然气就会被计算为“花费”，我不会收回。 矿工将停止处理交易，恢复所做的任何更改，但仍将其作为“失败的交易”包含在区块链中，收取费用。 这看起来可能很苛刻，但是当你意识到矿工真正的工作是在执行计算的时候，你可以看到他们永远也不会获得这些资源。所以，即使你设计糟糕的交易用完了，你付给他们的工作也是公平的。

提供太多的费用也不同于提供太多的 ether。 如果你设置了一个非常高的 gas price，那么你只需要付出很少的代价，就像在比特币中设置超高的交易费用一样。 你肯定会被排在最前面，但你的钱已经没有了。 但是，如果您提供了正常的 gas price，并且只需要支付比您购买 gas 所需的更多的 ether，那么超额部分将退还给您。 矿工只收取你实际工作的费用。 你可以把煤气价格看作矿工的小时工资，把煤气成本看作是工作时间表。

gas 是使以太坊中的复杂计算“安全”的关键机制，因为任何失控的程序只会在请求运行的人提供的资金的情况下持续下去。 当资金停止时，矿工们就停止工作。而你在程序中犯的错误只会影响付费使用它的人 - 网络的其他部分不会因为你的错误而遭受性能问题。 当性能问题消耗掉所有的 ether 时，他们只会得到一个大的薪水！ 如果没有这个关键技术，通用区块链的想法将是完全不可能的。

gas 是在以太坊进行的每一项操作的执行费用。 其价格用 ether 表示，由矿工决定，可拒绝低于一定的 gas price 处理。 为了得到 gas，你只需要添加

ether 到你的帐户。

以太坊在区块链上实施了一个名为以太坊虚拟机（EVM）的执行环境。 当您运行分散式应用程序（dApp）时，每个指令都会在网络的每个节点上执行。 这有一个代价：对于脚本可以执行的每个操作，都有一个指定的成本，用 gas 单位数表示，您可以在 EVM 规范中看到。

天然气价格由矿主决定，目前约为 5～21 GWei（1 GWei 为 10 ^ 9 Wei 或 10 ^ -9 Ether）。 以太坊使用以太币作为其内部货币/标记。 您的帐户持有以 ether 表示。 当您部署合同或执行交易时，gas 将从您的账户余额中提取。 你可以自由指定一个 gas price，或保留建议。

从理论上的 PoV 来看，每个采矿节点应该选择一个最大化其利润的天然气价格。 由于消耗更多天然气的块体在网络中传播速度较慢，因此它将成为一名叔叔的可能性较高，只能减少奖励。 接受的最低天然气价格应足够高，以应付这种增加的风险。 到目前为止，在真实网络中观察到的情况是，矿业集团接受交易，降低天然气价格，这在经济上是合理的，有助于减少网络拥塞并提高整体网络/代币价值。

gas 是以太坊“世界电脑”使用的计量单位。 作为比喻，电力按千瓦时计量。 在以太坊使用更多的计算和存储意味着使用更多的气体。 计量的一个根本原因是，它激励人们（矿工）操作世界计算机。 这些矿工获得处理交易的费用，这是由计量方案确定的：gas。

EVM 中的每个操作都会消耗 gas。 例如，乘法（MUL）消耗 5 个 gas，加法（ADD）消耗 3 个 gas。 以下是以太坊的运行和耗气量的电子表格。 （也可以把它们看作是天然气的成本，但是这可能使得解释更加难以跟随成本，费用，价格飞涨。）

计量是不同的费用和天然气是不同的乙醚。 为了澄清这个问题，把气体看成是燃料的代名词。 交易必须提供足够的燃料或启动燃料来覆盖 EVM 计算和存储设施的整个使用。 所有剩余的天然气将退还给交易的发起者：发起交易的用户。 耗尽气体的交易已经恢复，但仍然包含在一个区块中，并且相关费用支付给矿工。

从燃料的角度来看，我们来看一下收费的概况。 尽管 EVM 中的每个操作都消耗了预定义的固定 gas 量（例如，MUL 操作总是消耗 5 个气体），但是用户可以在每次交易中指定 gas price。 目前的天然气价格是 0.02μEther，或 0.00000002ETH。 发起人支付给矿工的费用是交易的（开始天然气 - 剩余天然气）× gas price。

4. EVM 中的数据存储结构

4.1. VM 内存

内存结构像堆栈一样，也提供了数据缓存的功能，但更作用的是提供了合约调用合约等过程中，子合约数据的临时存储。

1）实现类

```
public class Memory implements ProgramListenerAware
```

2）结构

```
private List<byte[]> chunks = new LinkedList<>();
```

内存就是一个链表。

链表的每个结点对应一个内存块，一个内存块 1024 字节，链块内存是可扩展的，扩展是以块为单位的。

private int softSize;

在内存中，有一个指针 softSize，用于指示存储数据的末尾。

3）外部接口

public byte[] read(int address, int size) {

从内存的 address（块抹平了的字节偏移量，从 0 开始，如 1025 代表第 1 块第 1 个位置）开始读取 size 个字节的数据。

address 可以超过存储数据的末尾。

读取时，如果被读数据超出了末尾指针的范围，则会以块为单位增加块数，也就是说读到的新增数据是 0。而末尾指针则以字，即 32 字节为单位移动，最终移动到被读数据的末尾。

public void write(int address, byte[] data, int dataSize, boolean limited)

将 data 的 dataSize 字节写到 address 上。

dataSize 如果大于 data 长度，则最多写 data 长度个字节。

limited 表示是否根据末尾指针截断，如果不，即不限制，则扩展内存，写入 dataSize 个字节，否则写入从 address 到末尾的字节个数。

public void extendAndWrite(int address, int allocSize, byte[] data) {

扩展然后写入数据。

public void extend(int address, int size) {

扩展内存，扩展了后末尾指针就变化。

public DataWord readWord(int address) {

从指定位置读取 32 个字节的字。

public byte readByte(int address) {

从指定位置读取一个字节。

public int size() {

获取末尾指针的位置。

public int internalSize() {

获取内存以分配空间，即链表结点数乘以 1024。

public List<byte[]> getChunks() {

获取内存副本。

public String toString() {

以打印格式（含 16 进制和 assii 码形式）返回内存数据，softSize 结尾。

4）程序监听

programListener.onMemoryExtend(toAllocate)

programListener.onMemoryWrite(address, data, dataSize)

在内存扩展（这里的扩展指应的字数）和写数据是会触发监听器。

4.2. VM 堆栈

EVM 的执行模型是基于栈结构的，像 JVM 一样，这里提供的堆栈就是用来存储字节码执行过程中的中间数据等。

1）实现类

public class Stack extends java.util.Stack<DataWord> implements ProgramListenerAware {

2）结构

直接继承自 Java 的 Stack：

protected Object[] elementData;

只是 item 是 DataWord，即 32 字节的字。

可以看到这是一个定长数组，在需要扩展时会创建新数组，然后浅拷贝元素到新数组。

以太坊栈深设置为 1024。

3）外部接口

public synchronized DataWord pop() {

弹出数据。

public DataWord push(DataWord item) {

入栈数据，返回就是 item

public void swap(int from, int to) {

交换两个位置的数据。

4）程序监听

programListener.onStackPop();

programListener.onStackPush(item);

programListener.onStackSwap(from, to);

4.3. VM 持久化存储

持久化存储主要是独立于区块存储之外，存储以太坊的账户、合约代码以及合约的状态。

4.3.1. 账户

org.ethereum.core.AccountState

可用来表示外部账户或合约账户。

含 nonce、余额、状态根、代码哈希。

private final BigInteger nonce

对于外部账户，nonce 表示从此账户发出的交易数量。

对于合约账户，nonce 表示此合约内创建的合约数量。

设计 nonce 的目的是为了防止重放攻击，也就是防止一个交易被多次执行，因为每执行一个交易时，库中该交易发送者账户的 nonce 就会增加 1。

默认值是从配置常量取出的 0。

private final BigInteger balance

账户的余额，以 Wei 为单位。默认为 0。

private final byte[] stateRoot

用于存储合约内容 Trie 结构的哈希根。

默认是 sha3(RLP.encodeElement(EMPTY_BYTE_ARRAY))。

private final byte[] codeHash;

合约代码的哈希值。

默认值是 sha3(EMPTY_BYTE_ARRAY)。

提供了 RLP 编码和解码方法，以及编码结果缓存。

只要代码哈希或 nonce 不是默认的，则认为合约是存在的。

所谓空账户，是指代码哈希、nonce、余额都是默认的。空账户是需要被删除的。

5. 群组架构的好处

FISCO BCOS 开源社区正式对外发布 FISCO BCOS 的 2.0 版，该版本在可扩展性、性能、易用性、隐私隔离等方面均取得突破性进展，其新增的群组架构方案，可以让企业间像拉微信群一样快速组链，大大降低维护难度和管理成本。

FISCO BCOS 2.0 新增了很多特性，统称为“一体两翼多引擎”，其能产生多大的燃动力助推联盟链应用落地。

FISCO BCOS 2.0 新增了群组架构，用于克服系统吞吐能力的瓶颈。

有别于传统区块链平台整个网络维护一个账本，所有节点参与到这个账本的共识和存储的做法，群组架构允许网络中存在多个不同的账本，每个账本是一个独立的小组，节点可以选择加入某些小组，参与到该组账本的共识和存储。该架构的特点是：

各群组独立执行共识流程，由群组内参与者决定如何进行共识，一个群组内的共识不受其他群组影响，各群组拥有独立的账本，维护自己的交易事务和数据，使得各群组之间解除耦合独立运作，可以达成更好的隐私隔离；

机构的节点只需部署一次，通过群组设置即可参与到不同的多方协作业务中，或将一个业务按用户、时间等维度分到各群组，群组架构可快速地平行扩展，在扩大了业务规模同时，极大简化了运维复杂度，降低管理成本。

6. 分布式存储有什么优势

（1）性能：在分布式存储达到一定规模是，性能会超过传统的 SAN、NAS。大量磁盘和节点，结合适当的数据分布策略，可以达到非常高的聚合带宽。传统的 SAN、NAS 都会有性能瓶颈，一旦达到最大扩展能力，性能不会改变甚至降低。

（2）价格：传统的 SAN、NAS，价格比较高。特别是 SAN 网络设备，光纤网络成本比较高。而且，以后扩展还需要增加扩展柜。成本太高。分布式存储只需要 IP 网络，几台 X86 服务器加内置硬盘就可以组建起来，初期成本比较低。扩展也非常方便，加服务器就行。

（3）可持续性：传统的 SAN、NAS 扩展能力受限，一个机头最多可以带几百个磁盘。如果想要个 PB 以上的共享存储，分布式存储只最好的选择。不用担心扩展能力问题。

7. 并行计算

并行计算（Parallel Computing）是指同时使用多种计算资源解决计算问题的过程，是提高计算机系统计算速度和处理能力的一种有效手段。它的基本思想是用多个处理器来协同求解同一问题，即将被求解的问题分解成若干个部分，各部分均由一个独立的处理机来并行计算。并行计算系统既可以是专门设计的、含有多个处理器的超级计算机，也可以是以某种方式互连的若干台的独立计算机构成的集群。通过并行计算集群完成数据的处理，再将处理的结果返回给用户。

并行计算可分为时间上的并行和空间上的并行。

时间上的并行：是指流水线技术，比如说工厂生产食品的时候步骤分为：

1. 清洗：将食品冲洗干净。
2. 消毒：将食品进行消毒处理。

3. 切割：将食品切成小块。

4. 包装：将食品装入包装袋。

如果不采用流水线，一个食品完成上述四个步骤后，下一个食品才进行处理，耗时且影响效率。但是采用流水线技术，就可以同时处理四个食品。这就是并行算法中的时间并行，在同一时间启动两个或两个以上的操作，大大提高计算性能。

空间上的并行：是指多个处理机并发的执行计算，即通过网络将两个以上的处理机连接起来，达到同时计算同一个任务的不同部分，或者单个处理机无法解决的大型问题。

比如小李准备在植树节种三棵树，如果小李 1 个人需要 6 个小时才能完成任务，植树节当天他叫来了好朋友小红、小王，三个人同时开始挖坑植树，2 个小时后每个人都完成了一颗植树任务，这就是并行算法中的空间并行，将一个大任务分割成多个相同的子任务，来加快问题解决速度。

8. 当前区块链实施的难度

首先，就是基础设施还相对不够完善。目前来看区块链已经是在各个行业都有所涉及，但对于区块链技术的基础设施来讲还是不够完善，其兼容性还没有达到特定的理想状态，去中心化以及安全隐私性做的还不够好，想要把区块链项目应用到实体经济体系当中就要把这些基础问题解决掉。

其次，新兴的区块链技术还是存在一定的技术缺陷的。在说到区块链技术的缺陷，明眼人心里都清楚，其交易的处理速度以及对资源利益的效率这些都是作为公链发展的阻碍，另外就是互联网的网速问题，再出现大量交易的时候会出现网络堵塞的问题。

再者，就是群龙无首。以区块链发展的趋势来看，其势头是有着不可阻挡的发展前景，区块链技术以其去中心化，数据信息不可篡改性以及溯源性让其成为了新兴技术中的老大。可是对于区块链技术想要在更多的行业生根发芽就需要行业的大佬以及大型企业的照顾，结合区块链技术对原有的营销模式进行升级改造，创造出一个全新的商业体系模式。只有有一个知名度高的企业应用区块链技术来打开市场，这样才有可能让区块链落地实施。

伴随着区块链行业慢慢的发展，其势头已经是有点势不可挡了，众多的科研人员也是马不停蹄的研究解决区块链的诸多疑点难题。小编相信只要解决掉最为基础的区块链技术的设施问题就可以实现区块链行业的落地实施。目前对于行业的发展也是需要区块链这样的技术来带动自己发展，并且满足行业发展的时候还要完善区块链技术的基础设施。

参考文献：

[1] Nakamoto, Satoshi. "Bitcoin: A Peer-to-Peer Electronic Cash System".

[2]Internet of Things, Blockchain and Shared Economy Applications[J]. Steve Huckle,Rituparna Bhattacharya,Martin White,Natalia Beloff. Procedia Computer Science.

[3]MacDonald T.J., Allen D.W.E., Potts J. (2016) Blockchains and the Boundaries of Self-Organized Economies: Predictions for the Future of Banking. In: Tasca P., Aste T., Pelizzon L., Perony N. (eds) Banking Beyond Banks and Money. New Economic Windows. Springer, Cham

[4] Christidis,Konstantinos,Michael Devetsikiotis.Blockchains and Smart Contracts for the Internet of Things [J],IEEE Access(Volume:4),2016.

[5]安瑞,何德彪,张韵茹,李莉. 基于区块链技术的防伪系统的设计与实现[J]. 密码学报,2017,4(02):199-208. DOI：10.13868/j.cnki.jcr.000174

[6]田海博,何杰杰,付利青. 基于公开区块链的隐私保护公平合同签署协议[J]. 密码学报,2017,4(02):187-198. DOI：10.13868/j.cnki.jcr.000173

[7]杨现民,李新,吴焕庆,赵可云.区块链技术在教育领域的应用模式与现实挑战[J/OL].现代远程教育研究,2017,(02):34-45.(2017-03-17).

[8]朱岩,甘国华,邓迪,姬菲菲,陈爱平.区块链关键技术中的安全性研究[J].信息安全研究,2016,2(12):1090-1097.

[9] Narayanan, Arvind; Bonneau, Joseph; Felten, Edward; Miller, Andrew; Goldfeder, Steven (2016). Bitcoin and cryptocurrency technologies: a comprehensive introduction. Princeton: Princeton University Press

[10] Becker, Georg (2008-07-18). Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis

[11] Antonopoulos, Andreas M. Mastering Bitcoin: Unlocking Digital Cryptocurrencies. Newton, MA: O’Reilly Media, 2014.