以太坊与智能合约第三章

郑嘉文

2020/09

讲师简介

- 贝尔大型电子商务网站架构师
- 贝尔大数据研发部门经理

- 华中理工大学计算机科学工程系
- 加拿大麦吉尔大学信息学硕士
- 加拿大多伦多大学罗特曼商学院MBA

多元文化背景
Multiple Culture

- 中文, 英文, 日文, 法文

多学科交叉背景
Multiple Disciplinary

- 计算机, 金融/CFA, 统计, 数学

多才多艺技术背景
Multiple Skill Set

- Java/J2EE, C#/.Net, C/C++, Python/Django, Go, Rust …

著作/Author

怎当她临去时秋波那一转

澹泊明志，宁静致远

帐号设置

2	9	0	0
积分	粉丝	关注	收藏夹

写博客　草稿箱 (1)

技能雷达

专长领域： WEB开发, 服务器端开发, CTO/CEO/CXO

开发平台： Java EE, C/C++, Python, .NET/C#, JavaScript

全部博文　热门博文　动态

原 **Solidity Event是如何实现的**

一个Solidity Event的定义如下： event De
indexed参数. 如果一个 indexed 参数的类

Solidity

BlockChain　08/07 16:54　4　0

原 **用智能合约实现金融期货**

比如你现在有10 ETH，今天价值$10,000 t
以你想保证2个月后你的账户里一定有$10,0

BlockChain　07/31 13:19　2　0

原 **FOMO游戏代码解析**

源代码在此处 智能合约代码部署在:
https://etherscan.io/address/0xa621428

BlockChain　07/21 17:39　20　0

技术博客： https://my.oschina.net/gavinzheng731/

免责声明

本系列课程所使用的例子程序仅仅用于教学目的。不得在实际的生产环境中使用。对于将本系列课程的例子程序应用于生产环境（真实环境）的行为以及引起之不良后果，**本人概不负责，特此声明。**

第一课目录/Contents

01 以太坊简介

02 以太坊关键数据结构

03 智能合约介绍

01

以太坊简介

以太坊/Ethereum

以太坊官方定义：

“以太坊是一个分布式的平台，可以运行智能合约：应用程序按照既定程序运行，不会出现停机，审查，欺诈或第三方干扰的可能性。这些应用程序运行在定制构建的区块链上，这是一个功能强大的全球共享基础架构，可以通过数字流转来代表财产的所有权。”

$$\sigma' = Y(\sigma, T)$$

状态机

去中心化

智能合约

智能合约/Smart Contract

智能合约顾名思义就是自动化合约。它们是自动执行的，并在其代码上写入了特定的指令，并在特定条件下执行。

智能合约是一系列指令，使用编程语言“ solidity” 编写，该编程语言基于IFTTT逻辑（即IF-THIS-THEN-THAT逻辑工作：如果符合某个条件则做某件事情）

智能合约编程语言： Solidity， Serpent， LLL

通常情况下，智能合约可以基于以下两种系统之一运行：

- 虚拟机：以太坊
- Docker：Fabric

EVM上运行的智能合约无法访问**网络，文件系统，或者在EVM上运行的其他进程**

燃料/GAS

GAS
你愿意为一个单位的Gas出多少Eth，一般用Gwei作单位

交易手续费(Tx Fee) = 实际运行步数(Actual Gas Used) * 单步价格(Gas Price)

GAS Limit
一次交易中愿意支付的燃料费的上限

- 消耗的Gas就已经超过你设定的Gas Limit，那么这次交易就会被取消
- 消耗的Gas未达到Gas Limit，那么只会按实际消耗的Gas 收取交易服务费

为什么需要燃料

金融上的目的就是激励矿工去使用他们自己的时间和能源去执行交易和智能合约

燃料可以用来兼顾在网络里所有参与者的利益。部分地解决了在没有信任的环境里如何减轻有害的或者恶意攻击问题。

燃料最后是会被耗尽的,有问题的交易会因为燃料耗尽而退出，进而执行下一个交易。这保证了以太坊不会死机

- 经典的停机理论

其他

账户

外部账户: 该类账户被公钥-私钥对控制（人类）

合约账户:该类账户被存储在账户中的代码控制

地址公式 *A* =采用公钥的Keccak-256散列的最后40个字符/ 20个字节（Keccak-256）

长度 160bit = 20 Byte

例子 1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy

交易

就是被外部拥有账户生成的加密签名的一段指令，序列化，然后提交给区块链。

有两种类型的交易：消息通信和合约创建(也就是交易产生一个新的以太坊合约)。

EVM

EVM是所有智能合约在以太坊中运作的虚拟机器。它是一个简单而强大的图灵完备的 256位虚拟机。

图灵完备意味着在给定资源和内存的情况下，EVM中执行的任何程序都可以解决任何问题

02

以太坊关键数据结构

递归长度前缀编码(Recursive Length Prefix，RLP)提供了一种适用于任意二进制数据数组的编码，RLP是以太坊中对对象进行序列化的主要编码方式。 RLP的唯一目标就是存储嵌套的字节数组

RLP 编码函数接受如下两种数据类型：

- 字符串
- 列表

递归长度前缀编码规则

- 对于 [0x00, 0x7f(127)] 范围内的单个字节, RLP 编码内容就是字节内容本身。
- 否则，如果是一个 0-55 字节长的字符串，则RLP编码有一个特别的数值 0x80(128) 加上字符串长度，再加上字符串二进制内容。这样，第一个字节的表达范围为 [0x80(128), 0xb7(183)].

  rlp编码=0x80+hex(len(str))+str

- 如果字符串长度超过 55 个字节，RLP 编码由定值 0xb7 加上字符串长度所占用的字节数，加上字符串长度的编码，加上字符串二进制内容组成。比如，一个长度为 1024 的字符串，将被编码为\xb9\x04\x00 后面再加上字符串内容。第一字节的表达范围是[0xb8(184), 0xbf(191)]。

  rlp编码=0xb7+hex(len(bin(len(str))))+bin(len(str))+str

- 如果列表的内容（它的所有项的组合长度）是0-55个字节长，它的RLP编码由0xC0加上所有的项的RLP编码串联起来的长度得到的单个字节，后跟所有的项的RLP编码的串联组成。 第一字节的范围因此是[0xc0(192), 0xf7(247)]

  rlp编码= 0xc0 + len(list) + rlp(list_item1) + rlp(list_item2) + ... + rlp(list_itemn)

- 如果列表的内容超过55字节，它的RLP编码由0xC0(192)加上所有的项的RLP编码串联起来的长度的长度得到的单个字节，后跟所有的项的RLP编码串联起来的长度的长度，再后跟所有的项的RLP编码的串联组成。 第一字节的范围因此是[0xf8(248), 0xff(255)] 。

  rlp编码= 0xf7 + hex(len(bin(len(list)))) + bin(len(list)) + rlp(list_item1) + rlp(list_item2)...+rlp(list_itemn)

前缀树或字典树，是一种有序树，用于保存关联数组，其中的键通常是字符串。与二叉查找树不同，键不是直接保存在节点中，而是由节点在树中的位置决定。一个节点的所有子孙都有相同的前缀，也就是这个节点对应的字符串，而根节点对应空字符串

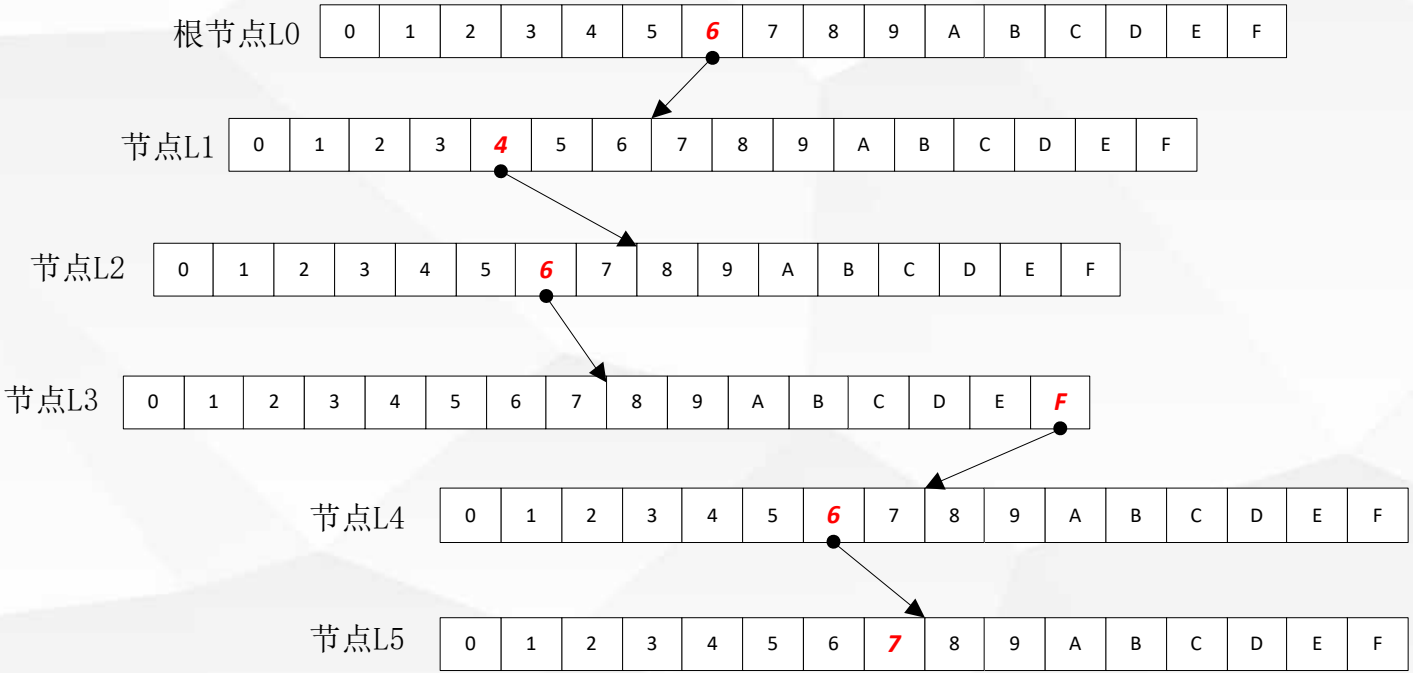
基数树的优势在于相比于哈希表，使用前缀树来进行查询拥有相同前缀key的数据时十分高效。对于最差的情况(前缀为空串)，复杂度为O(n)，仍然需要遍历整棵树，此时效率与哈希表相同。相对于哈希表，在前缀树不会存在哈希冲突的问题。

基数树主要的缺陷：

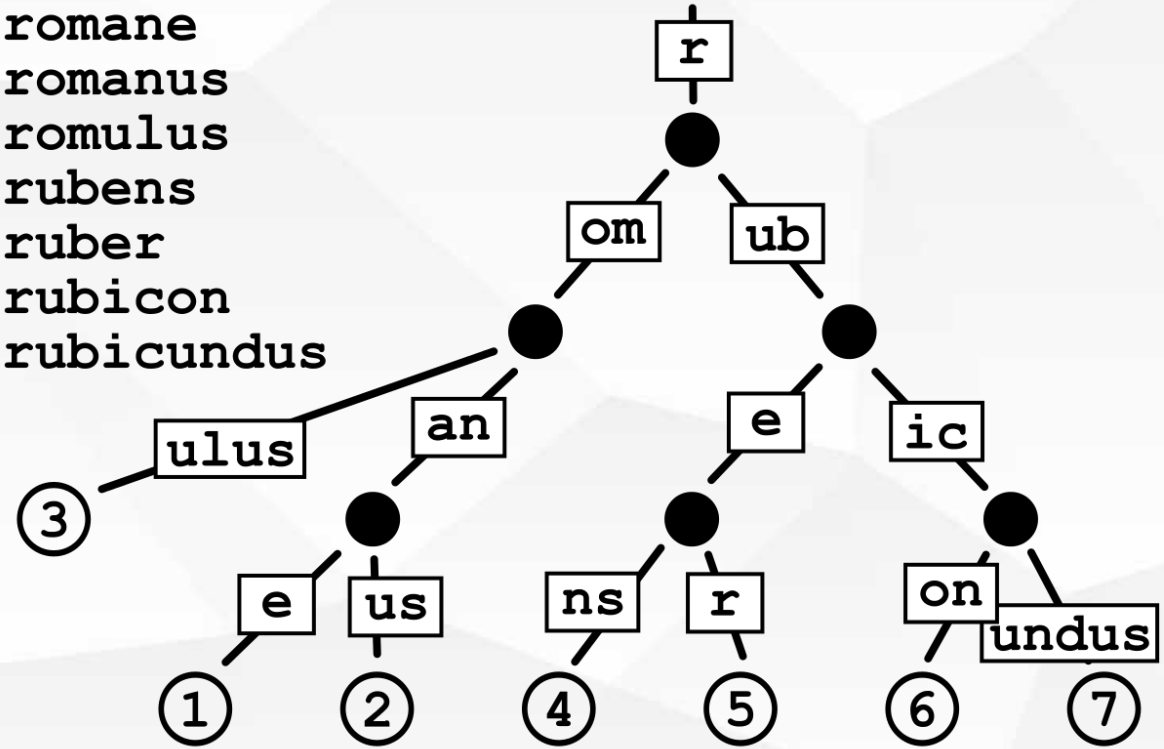
- 直接查找效率低下
  前缀树的查找效率是O(m),m为所查找节点的key长度，而哈希表的查找效率为O(1)。且一次查找会有m次IO开销，相比于直接查找，无论是速率、还是对磁盘的压力都比较大。
- 低效

前缀树 - 例子

例： “dog” 其其ASCII码值为646f67



```
1 romane
2 romanus
3 romulus
4 rubens
5 ruber
6 rubicon
7 rubicundus
```



默克尔帕特里夏树MPT

梅克尔帕特里夏树就是Merkle Tree和Patricia Tree两者混合后的产物。MPT树有以下几个作用：

- 存储任意长度的key-value键值对数据
- 提供了一种快速计算所维护数据集哈希标识的机制
- 提供了快速状态回滚的机制
- 提供了一种称为***默克尔证明***的证明方法，进行轻节点的扩展，实现简单支付验证

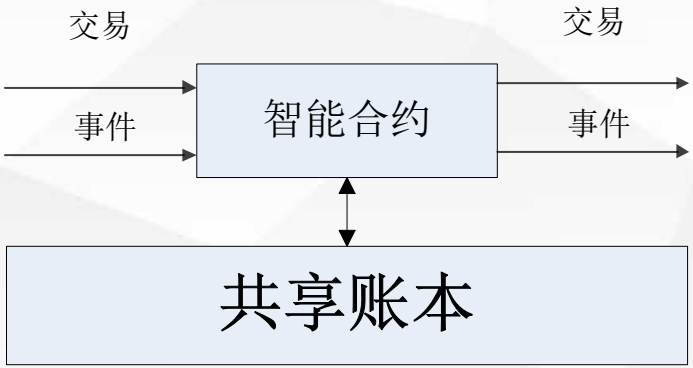
03

智能合约介绍

尼克萨博（Nick Szabo） *“一个智能合约是一套以数字形式定义的承诺（promises），包括合约参与方可以在上面执行这些承诺的协议。”*

智能合约可以基于以下两种系统之一运行：

- 虚拟机：以太坊
- Docker：超级账本Fabric



智能合约的2种实现方式

虚拟机 – 以太坊
- 高耦合方式，业务逻辑与虚拟机的强绑定
- 移植性差
- 升级困难

容器方式 – 超级账本
- 松耦合
- 容器的启动速度制约了执行效率
- 执行速度快

指令的执行速度

执行环境上下文本身的启动速度

现有智能合约框架

框架名	简介	类型	平台
LLL	具有Lisp语法的一个函数型编程语言 是第一个以太坊上的智能合约编程语言，不过现在很少用了	声明性（Declarative）	以太坊
Serpent	具有类Python语法的过程性编程语言	描述型（Imperative）	以太坊
Solidity	具有类Javascript, C++或者Java语法的过程性编程语言，是目前最流行的智能合约编程语言	描述型（Imperative）	以太坊
Vyper	类似Serpent，采用Python类似的语法，最近开发的编程语言	描述型（Imperative）	以太坊
Bamboo	受Erlang语言影响，新开发的智能合约编程语言，有显式的状态变换，没有循环。		以太坊
WASM	**可以用任何流行的编程语言来编写智能合约，只要能够编译成WASM格式即可。WASM被认为是智能合约编程语言领域内的明日之星**	**描述型（Imperative）**	**通用**
Golang/Nodejs	可以用Golang和Nodejs等高级编程语言来编写智能合约	描述型（Imperative）	超级账本

谢谢聆听

郑嘉文

2018/08