软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2019,30(9):2571−2592 [doi: 10.13328/j.cnki.jos.005775]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

企业级区块链技术综述*

邵奇峰[1,2], 张 召[1], 朱燕超[1], 周傲英[1]

[1](华东师范大学 数据科学与工程学院,上海 200062)

[2](中原工学院 软件学院,河南 郑州 450007)

通信作者: 张召, E-mail: zhzhang@dase.ecnu.edu.cn

摘 要: 在传统跨机构交易的企业应用中,各个机构都是独立记录己方的交易数据,机构间数据的差异会引起争议,通常需要人工对账或中介机构来解决,因而增加了结算时间和交易费用.区块链技术实现了交易数据在写入前共识验证、写入后不可篡改的分布式记账,可信地保证了多机构间的数据一致性,避免了人工对账和中介机构.区块链是一种去中心化、不可篡改、可追溯、可信的、多方共享的分布式数据库,企业级区块链是节点加入需经许可的适用于企业级应用的区块链技术.结合 Hyperledger Fabric,Corda 和 Quorum 等企业级区块链平台,提出了企业级区块链的系统架构;从交易流程、区块链网络、共识机制、区块链数据、智能合约、隐私保护几方面阐述了企业级区块链的原理与技术;针对企业级区块链的现状,总结了当前的研究挑战与未来的发展趋势.

关键词: 企业级区块链;Hyperledger Fabric; Corda; Quorum

中图法分类号: TP311

中文引用格式: 邵奇峰,张召,朱燕超,周傲英.企业级区块链技术综述.软件学报,2019,30(9):2571−2592. http://www.jos.org.cn/1000- 9825/5775.htm

英文引用格式: Shao QF, Zhang Z, Zhu YC, Zhou AY. Survey of enterprise blockchains. Ruan Jian Xue Bao/Journal of Software, 2019,30(9):2571−2592 (in Chinese). http://www.jos.org.cn/1000-9825/5775.htm

Survey of Enterprise Blockchains

SHAO Qi-Feng[1,2], ZHANG Zhao[1], ZHU Yan-Chao[1], ZHOU Ao-Ying[1]

[1](School of Data Science and Engineering, East China Normal University, Shanghai 200062, China)

[2](Software College, Zhongyuan University of Technology, Zhengzhou 450007, China)

Abstract: In legacy enterprise applications of cross-institution transactions, all institutions maintain their own ledgers. The discrepancies between different ledgers result in disputes and increase the need for manual reconciliations with settlement times and intermediaries with associated overhead costs. However, a blockchain implementing a distributed ledger, where transactions must be validated by consensus and cannot be altered once written to the ledger, guarantees the consistency of multi-institutional data and removes manual reconciliations and intermediaries. Blockchain is a decentralized, tamper-proof, traceable, trustless distributed database managed by multiple participants. An enterprise blockchain satisfying enterprise application requests means that any node must be authorized and authenticated in order to join the network. This paper presents an architecture model of enterprise blockchains based on the three mainstream blockchain platforms: Hyperledger Fabric, Corda, and Quorum. Furthermore, the principles and technologies of enterprise

* 基金项目: 国家自然科学基金(61432006, 61672232, 61332006); 国家高技术研究发展计划(863)(2015AA015307); 河南省科技攻关计划(172102310714, 172102210593); 河南省高等学校重点科研项目(15A520112)

Foundation item: National Natural Science Foundation of China (61432006, 61672232, 61332006); National High Technology Research and Development Program of China (863) (2015AA015307); Science and Technology Program of He'nan Province (172102310714, 172102210593); Colleges and Universities Key Research Project of He'nan Province (15A520112)

本文由"区块链数据管理"专题特约编辑于戈教授、牛保宁教授、金澈清教授推荐.

收稿时间: 2018-06-10; 修改时间: 2018-08-28; 采用时间: 2018-12-14; jos 在线出版时间: 2019-04-10

CNKI 网络优先出版: 2019-04-09 17:32:22, http://kns.cnki.net/kcms/detail/11.2560.TP.20190409.1732.004.html

blockchains according to transaction flow, P2P network, consensus mechanism, blockchain data, smart contract, and privacy are discussed. Finally, by analyzing the limitations of the existing technologies, some challenging research issues and technology trends of enterprise blockchains are summarized.

Key words: enterprise blockchain; Hyperledger Fabric; Corda; Quorum

在涉及多方交易的场景中,目前的企业级应用各自记录己方的交易数据,不同交易方多个账本间的数据差异会引起分歧与争议,因而需要人工对账或第三方中介裁决,增加了交易的延迟和费用.例如,目前美国股票交易时间不到 1 秒,但在证券存托与清算公司(depository trust & clearing corporation,简称 DTCC)的结算时间却为 3 天,限制了资本流动性[1].各国银行机构主要通过环球银行金融电信协会(society for worldwide interbank financial telecommunications,简称 SWIFT)实现跨币种、跨境交易,目前每笔交易需支付百元左右的电讯费,且转账时间为 2 天~3 天[2].区块链(blockchain)技术实现了多方共享的全局性单一账本,解决了多方独立记账所带来的数据不一致性问题.区块链是一种去中心化、不可篡改、可追溯、可信的分布式数据库,由互不信任的多方参与者共同维护;每笔交易需经全网大多数参与者验证且达成共识后,才会一致性地记录在全网节点;交易一旦被记录,任何人都无法篡改,从而实现了可信的多方间数据共享,避免了人工对账,消除了中介机构,减少了交易的延迟与费用.区块链是不同于传统数据库的新型数据库,表 1 对比了区块链与传统数据库的差异.

Table 1 Comparison of blockchain and DBMS

表 1 区块链与 DBMS 对比

	区块链	DBMS
应用	可信的企业间数据管理	高效的企业内数据管理
服务端程序	智能合约	存储过程
共识机制	BFT 共识	CFT 共识
中心化	去中心	强中心
网络	节点间互不信任的对等式网络	节点间互相信任的主从式网络
访问控制	基于数字签名的身份认证	基于用户与角色的权限管理
交易存储	不可篡改、可追溯的区块链	支持高速写入的预写式日志
数据结构	查询可验证的 Merkle 树	存取高效的 B-树
数据库管理员	无控制全局的管理员	控制全局的管理员

2008 年,化名为中本聪(Satoshi Nakamoto)的学者提出了比特币,这是一种无需任何权威机构背书的去中心化的数字货币支付系统[3].随后人们发现了从比特币底层提取出的区块链技术不仅能够应用于数字货币,还能够使用户在无需相互信任与可信中介的场景下实现可信的价值传输.为此出现了一批用以太坊(ethereum)[4]为代表的实现数字资产交易的区块链平台.任何节点无需许可就能够随时加入/退出,所以此类区块链被称为公有链(public blockchain)或非许可链(permissionless blockchain).公有链允许任何节点随意进出的特性显然无法适用于企业级应用.在跨机构的交易场景中,相互协作的多家企业组成联盟,只有联盟成员才可加入区块链及参与交易.此类节点需经许可才能加入的区块链,被称为企业级区块链(enterprise blockchain)、联盟链(consortium blockchain)或许可链(permissioned blockchain).公有链和企业级区块链针对的应用场景不同、解决的问题领域不同,它们之间的主要区别见表 2.

Table 2 Comparison of public blockchain and enterprise blockchain

表 2 公有链与企业级区块链对比

	公有链	企业级区块链
节点准入	节点自由加入	节点需经许可才能加入
用户管理	任何用户均可加入,用户身份匿名	用户需经身份核实才可加入,用户身份实名
去中心化	部署在全球范围,实现的是完全去中心化	部署在企业联盟内,实现的是弱中心化
共识机制	采用 PoW[3]、PoS[5]等基于证明的共识机制	采用 PBFT[6]、Raft[7]等基于投票的共识机制
数字货币	发行数字货币,激励更多节点参与记账和运营	为实现企业间业务而构建,无需发行数字货币与激励
节点数量	节点数<30000	节点数<100
交易存储	每个节点全量存储着全网交易数据	因涉及商业机密,各个节点一致性地存储与自己业务相关的交易数据和其他方交易数据的哈希

基于比特币、以太坊等公有链的成功经验及企业级应用需求,业界推出了一批支持企业级应用的区块链平台.2015 年 12 月,Linux 基金会发起了 Hyperledger(https://www.hyperledger.org)开源区块链项目,着重于发展跨行业的企业级区块链.Hyperledger 分别提出了 Fabric(https://github.com/hyperledger/fabric),Sawtooth(https://github.com/hyperledger/sawtooth-core),Iroha(https://github.com/hyperledger/iroha),Burrow(https://github.com/hyperledger/burrow)和 Indy(https://github.com/hyperledger/indy-node)等多个企业级区块链平台,以适应不同的需求和场景.Hyperledger Fabric 应用最为广泛,其采用了合约执行与共识机制相分离的系统架构,模块化地实现了共识服务、成员服务等服务的即插即用.Hyperledger Sawtooth 基于 Intel SGX(software guard extensions)[8]可信硬件实现了经历时间证明(proof of elapsed time,简称 PoET)[9]共识机制,相对于 PoW 共识,其无需挖矿且出块间隔更短.Hyperledger Iroha 主要针对于移动应用,其实现了基于链复制(chain replication)[10]的共识机制 Sumeragi.Hyperledger Burrow 集成了以太坊虚拟机并可运行以太坊智能合约,其使用了 Tendermint[11]共识机制.Hyperledger Indy 是基于区块链的去中心的数字身份平台,其使用了 RBFT(redundant byzantine fault tolerance)[12]共识机制.2016 年 4 月,R3 金融区块链联盟(https://www.r3.com)提出了 Corda[13,14]平台,着重服务于受监管的金融行业,强调业务数据仅对交易双方及监管可见的数据隐私性,反对数据全网广播及每个节点拥有全部数据.Corda 自称是受到区块链启发的分布式账本[15],在技术架构上有许多特色与创新.2016 年 9 月,摩根大通提出了基于以太坊构建的企业级区块链平台 Quorum[16],其通过分别处理公有交易和私有交易实现了交易和合约的隐私保护,并用 Raft 共识替换了以太坊的 PoW 共识.2017 年 2 月,企业以太坊联盟(enterprise ethereum alliance,简称 EEA)(https://entethalliance.org)成立,旨在合作开发标准和技术以拓展以太坊适用于企业级应用,Quorum 即是 EEA 的技术参考实现.Chain Core(https://github.com/chain/chain)是由 Chain 公司提出的企业级区块链平台,主要专注于金融行业的数字资产服务,其基于 Chain Protocol 实现了资产的发行、传输和控制.MultiChain[17]是由 Coin Sciences 公司提出的企业级区块链平台,其兼容于比特币系统,侧重于数字资产类应用,可快速部署在 Windows,Linux 和 Mac OS 多种操作系统之上.Ripple[18]是瑞波公司提出的基于分布式账本的实时跨境支付网络,其通过 ILP(interledger protocol)[19]协议实现了不同账本与支付系统间的互联.BigchainDB[20]是由 BigchainDB 公司提出的可扩展的区块链数据库,其声称既拥有高吞吐量、低延迟、大容量、丰富查询和权限等分布式数据库的优点,又拥有去中心化、不可篡改、资产传输等区块链的特性,因此被称为在分布式数据库中加入了区块链特性.2017 年 7 月,微众银行、万向区块链和矩阵元联合提出了开源企业级区块链平台 BCOS(https://github.com/bcosorg/bcos),为了适用于企业级应用,其在以太坊基础上加入了 CA 身份认证、PBFT 共识机制、隐私保护等组件,在国内率先应用于金融领域并取得了商用实践成果.随后,又联合金链盟提出了着重于解决金融行业高频交易、安全性及合规方面需求的 BCOS 分支版本 FISCO BCOS(https://github.com/FISCO-BCOS).表 3 分别从数据模型、共识机制、智能合约语言、智能合约沙箱、底层数据库几个方面对比了以上企业级区块链平台.

Table 3 Comparison of enterprise blockchain platform

表 3 企业级区块链平台对比

	数据模型	共识机制	智能合约语言	智能合约沙箱	底层数据库
Hyperledger Fabric	基于账户	Solo/Kafka/PBFT	Go/Node.js/Java	Docker	LevelDB/CouchDB
Hyperledger Sawtooth	基于账户	PoET	Transaction Family	–	–
Hyperledger Iroha	基于账户	Sumeragi	–	–	PostgreSQL
Hyperledger Burrow	基于账户	Tendermint	Solidity	EVM	LevelDB
Hyperledger Indy	基于账户	RBFT	–	–	RocksDB
Quorum	基于账户	Raft/PBFT	Solidity	EVM	LevelDB
Ripple	基于账户	RPCA	–	–	SQLite/RocksDB
BCOS	基于账户	Raft/PBFT	Solidity	EVM	LevelDB
Corda	基于交易	Notary(单节点/Raft/PBFT)	Java/Kotlin	JVM	常用关系数据库
Chain Core	基于交易	Federated Consensus	Ivy	CVM	PostgreSQL/RocksDB
Multichain	基于交易	Randomised Round-robin	–	–	LevelDB
BigchainDB	基于交易	Majority Voting	Crypto-Conditions	–	RethinkDB/MongoDB

企业级区块链产品众多,但最有影响力的是 Linux 基金会的 Hyperledger Fabric、R3 联盟的 Corda 和 EEA 参考实现 Quorum.目前,Hyperledger 包括 IBM、Intel、百度等 200 多家成员;R3 联盟包括花旗银行、汇丰银行、德意志银行等 200 多家以金融机构为主的成员;EEA 包括摩根大通、微软、Intel 等 400 多家成员.Fabric,Corda 和 Quorum 平台都有着完善的软件实现、广泛的用户群体和充分的运营实践,产品版本都在 1.0 之上且系统代码开源.文献[21–23]主要基于比特币介绍了区块链技术的研究现状,文献[24–26]从数据处理的角度综述了区块链技术.与已有文献不同,本文主要结合 Fabric,Corda 和 Quorum 平台的共性与差异进行对比分析,在此基础上,介绍了企业级区块链技术的研究现状与发展趋势.本文第 1 节提出企业级区块链的系统架构.第 2 节从整体结构上描述 Fabric,Corda 和 Quorum 的交易流程.第 3 节从网络层阐述准入机制和网络协议.第 4 节介绍共识机制的实现方案.第 5 节对比分析 Fabric,Corda 和 Quorum 的数据组织与结构.第 6 节从合约模型、沙箱环境论述智能合约.第 7 节分别介绍 Fabric、Corda 和 Quorum 的隐私保护方案.第 8 节总结企业级区块链的研究挑战与趋势.结束语展望发展中的企业级区块链.

1 系统架构

各类企业级区块链尽管面向的应用领域不同,具体设计实现细节不同,但在整体系统架构上还是存在着诸多共性.如图 1 所示,企业级区块链可划分为成网络层、共识层、数据层、智能合约层和应用层.

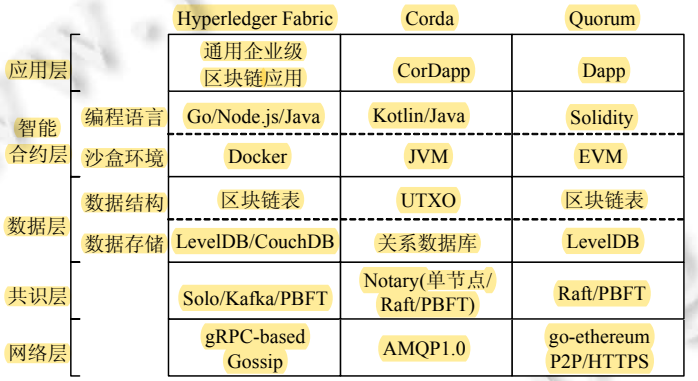

Fig.1 Architecture of enterprise blockchain

图 1 企业级区块链系统架构

1.1 网络层

区块链强调网络去中心化,强调节点平等、自治.以比特币、以太坊为代表的公有链允许任何节点随意加入退出且身份匿名,所以采用 P2P 协议广播消息以实现路由发现、节点识别、传播交易数据与区块数据[27].对于企业级区块链而言,所有节点经审核后才可加入网络且身份已知,其交易数据因涉及商业机密而不宜全网广播,因此就需要企业级区块链在考虑去中心化的同时,既需加入节点准入机制,还要利用节点身份已知、数据传播范围受限等特性对网络协议进行优化.Fabric 采用了基于 gRPC 的 Gossip[28,29]协议,且支持 TLS 加密通信.Corda 采用了 AMQP1.0[30]协议,且支持 TLS 加密通信.Quorum 分别采用以太坊 P2P 协议和 HTTPS 协议传输公有交易和私用交易.

1.2 共识层

比特币采用 PoW 共识机制是为了适应其节点身份匿名、自由进出、数目众多等公有链特性[31],但其带来的消耗计算资源、交易未最终确认、交易吞吐量受限等问题,则是企业级区块链所无法接受的.企业级区块链主要采用 PBFT 共识机制,在 PBFT 共识过程中,有两个阶段需要传输的网络消息数为 $O(n^2)$,其造成了很大的网

络开销;另外,PBFT 共识假设网络中共识节点数目是静态不变的,这就影响到了共识服务的可扩展性.因此,企业级区块链通常实现的是各种改进的 PBFT 共识机制.

Kwon 等人[11]提出了 Tendermint,在基于 PBFT 按节点计票的基础上,对每张投票分配了不同的权重,重要节点的投票可分配较高的权重,若投票权重超过总数 2/3,即认为达成共识.仅通过少数重要节点达成共识,会显著减少网络中广播的消息数;在基于数字货币的应用中,权重也可对应为用户的持币量,从而实现类似权益证明的共识机制.Iroha 采用基于链复制的共识机制 Sumeragi,其源自于 Duan 等人[32]提出的 BChain 共识.BChain 将网络节点按信誉由高到低组织成逻辑上的链式结构,交易依照节点的链接顺序线性地进行传播.假设节点总数为 $3f+1$,正常情况下只需链中前 $2f+1$ 的节点参与共识.仅当前 $2f+1$ 的节点出现错误,才需要剩余 f 个节点参与共识.BChain 减少了网络中广播的消息数,但增加了传播延迟和节点出错后的恢复时间.Aublin 等人[12]提出了 RBFT,多台共识主机上并行运行着多组 PBFT 共识,每组的主节点都位于不同的主机.多组共识同时对请求进行排序,但仅有一组提交数据,若该组出现错误而性能降低时,则可迅速切换至其他组来提交数据.Chain Core 使用的共识机制为 Federated Consensus(https://chain.com/docs/1.2/protocol/papers/whitepaper),其假设创建区块的主节点是固定的、不作恶的、不宕机的,从而减少了共识中广播的消息数目.MultiChain 模拟了 PoW 共识机制,其预先设定多个可信节点,若某节点等待系统设定的随机时间率先到达,就可创建区块.为了保证出块节点的多样性,还可灵活配置节点需等待多少连续区块后才可创建下一区块,MultiChain 的共识机制也存在着分叉带来的交易没有最终确认的问题.Ripple 提出了仅由可信验证节点投票的共识机制 RPCA(ripple protocol consensus algorithm)[18],在 $n \geq 5f+1$ 的条件下,解决了拜占庭将军问题.此外,在假设节点不会作恶的前提下,很多平台还提供了吞吐量相对较高、实现相对简单的 Raft 共识机制.Fabric 提供了单节点的 Solo(供开发使用)、高吞吐量的 Apache Kafka[33]和基于 PBFT 的 BFT-SMaRt[34]这 3 种共识服务.Corda 提供了高信任场景下的单节点 Notary[13]服务,为了实现分布式的 Notary 服务,还提供基于 Raft 的 Copycat(http://atomix.io)和基于 PBFT 的 BFT- SMaRt 分布式共识.Quorum 提供了基于 Raft 的 etcd(https://github.com/coreos/etcd)和基于 PBFT 的 Istanbul BFT (https://github.com/ethereum/EIPs/issues/650)共识服务.

1.3 数据层

区块链系统通常将一批交易打包进区块,然后以区块为单位对这些交易进行共识、存储及执行.以区块为单位组织交易数据,并以区块哈希作为指针将孤立的区块按时间顺序链接在一起,一方面保证了交易数据的不可篡改性,另一方面也实现了交易数据的可追溯性[35].区块链数据模型可分为基于交易的模型和基于账户的模型[36]:基于交易的模型也被称为 UTXO 模型,其每笔交易由表明交易来源的输入和表明交易去向的输出组成,所有交易通过输入与输出链接在一起,使得每一笔交易都可追溯,避免了伪造与双花(double spending)问题,更适于支持数字资产;基于账户的模型维护着账户各属性的当前状态,通过执行交易来不断更新账户数据,更适于支持智能合约.区块链实现了交易数据按时间顺序的追溯,UTXO 模型则实现了交易数据按交易顺序的追溯.Fabric,Quorum 是基于账户模型的,其采用区块链来组织交易数据;Corda 是基于交易模型的,其采用 UTXO 模型来组织交易数据.在数据存储上,Fabric,Quorum 为了实现基于哈希的快速检索,选用了健值数据库 LevelDB,Fabric 还支持 CouchDB[37]数据库;Corda 考虑与企业已有系统的整合,选用了通用的关系数据库.此外,业界提出一些用图来组织交易数据的方案.IOTA(http://iota.org)平台提出了使用 DAG(directed acyclic graph)来组织交易的方案 Tangle[38],每个交易需至少链接之前的两个交易(即包含两个交易的哈希)以表示确认了这两个交易.Swirlds(https://www.swirlds.com)平台提出使用 Hashgraph[39]来组织交易的方案,其依靠节点间的 Gossip 通信历史构造了基于交易的哈希有向无循环图.

1.4 智能合约层

1994 年,Szabo 提出了智能合约[40],其被定义为一套以数字形式定义的承诺,包括合约参与方执行这些承诺所需的协议,其初衷是将智能合约内置到物理实体,以创造各种灵活可控的智能资产.智能合约最初并未受到广泛关注,直到以太坊将智能合约应用于区块链系统,使得区块链可以处理数字货币之外更为复杂的应用.智能合

约是一种用程序来编制的数字合约条款,部署在区块链上且可按照规则自动执行.区块链实现了去中心化的存储,智能合约作为运行在区块链上的商业逻辑,实现了去中心化的计算.智能合约可以是简单的状态数据修改、交易合规验证,也可以是依据商业合约及外部事件的复杂交易逻辑.为了保证在任意节点上智能合约的执行结果始终一致,智能合约通常运行在沙箱环境中.Fabric 智能合约运行在 Docker 容器,支持 Go,Node.js 和 Java 语言.Corda 智能合约运行在 JVM(Java virtual machine),支持 Kotlin 和 Java 语言.Quorum 智能合约运行在 EVM (ethereum virtual machine),支持 Solidity 语言.Chain Core 的智能合约运行在 CVM(Chain virtual machine),支持 Ivy 语言.Sawtooth 提出了交易家族(transaction family)的概念,其支持 Go,JavaScript,Python 等多种常用的编程语言,限定仅能执行特定的业务操作.Sawtooth 也整合了 EVM,以支持由 Solidity 编写的智能合约.

1.5 应用层

应用层通过调用智能合约,并依据合约中定义的规则执行交易.Fabric 主要面向通用行业,其应用可基于 Go,Java,Python,Node.js 等多种语言的 SDK 构建,并通过 gPRC 与运行在 Fabric 节点上的智能合约进行通信[41].Corda 主要面向受监管的金融行业,其应用被称为 CorDapp(Corda distributed application)[13],可基于 Kotlin 和 Java 语言构建.CorDapp 主要由状态对象、智能合约和 Flow[13]构成,Flow 定义了交易流程,在 Flow 执行过程中,需验证交易的状态对象符合智能合约定义的约束后,才签名并提交交易.Quorum 主要面向金融行业,与以太坊应用一致,其应用被称为 Dapp(decentralized application).Dapp 是由 JavaScript 构建的 Web 前端应用,通过 JSON-RPC 与运行在 Quorum 节点上的智能合约进行通信[42].

2 交易流程

区块主要包含的是交易数据,智能合约主要处理的也是交易数据,交易是区块链中存储和执行的基本逻辑单元.完整的交易流程可动态展示出区块链系统内部各组件间是如何协作的.本节分别介绍 Fabric,Corda 和 Quorum 的交易流程.

2.1 Fabric交易流程

Fabric 着重于模块化的架构设计,把系统分为了背书节点(endorsing peer)、排序服务(ordering service)和提交节点(committing peer)[43]:背书节点主要执行智能合约,排序服务主要执行共识以对交易排序并生成区块,提交节点主要持久化区块数据和状态数据.模块化设计实现了侧重于计算的合约执行、侧重于通信的网络共识和侧重于 I/O 的区块存储等各模块间的相互解耦,使得各种服务均可独立的横向扩展.如图 2 所示,Fabric 执行交易的完整流程如下.

(1) 客户端对新的交易数据签名并发送到一至多个背书节点;
(2) 背书节点以交易数据为输入执行智能合约并生成读写集(*readset*,*writeset*);
(3) 背书节点对读写集进行签名并返回至客户端;
(4) 客户端收集读写集,验证符合背书策略(endorsement policy)后将其广播至排序服务;
(5) 排序服务基于共识机制对多笔交易的读写集排序并将其打包成区块;
(6) 排序服务将区块传播至提交节点;
(7) 提交节点对从排序服务收到的区块中的读写集进行背书策略验证和读集(*readset*)版本验证,验证通过后,将区块追加至区块链,并将写集(*writeset*)写入状态数据库.

传统区块链平台采用的是的先排序、后执行的主动复制(active replication)模型[44],如果智能合约中含有非确定性(non-determinism)代码[45],就会造成节点间的数据不一致从而发生分叉.Fabric 提供的是一种先执行、后排序、再验证的系统架构,这实际上是一种结合被动复制(passive replication)[46]与主动复制的混合模型.如果多个背书节点返回的合约执行结果不一致,Fabric 可在数据写入之前就识别出非确定性,从而避免了不一致数据的写入.不同于传统区块链的智能合约串行运行在全部节点,Fabric 智能合约可并行运行在不同的背书节点,从而提高了系统交易吞吐量[47].同时,随着背书节点数目的动态增加,系统内可并行运行的合约数目也将随之增

加,从而实现了一定的可扩展性.从本质上来讲,传统区块链是将交易数据传播到所有节点,并通过在所有节点上执行而达到状态一致.Fabric 仅在部分可信的背书节点上执行交易,然后将执行结果传播到所有节点从而达到状态一致.

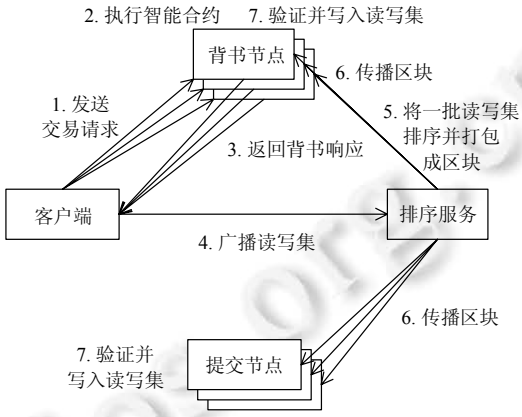

Fig.2 Transaction flow in Hyperledger Fabric

图 2 Hyperledger Fabric 交易流程

2.2 Corda交易流程

Corda 着重服务于受监管的金融行业,其强调交易数据仅对交易双方及监管可见.基于 P2P 协议广播交易数据难于控制数据的传播范围,所以 Corda 交易都是直接被传送至指定目的节点,Corda 网络中的共识服务则主要由 Notary 实现.如图 3 所示,Corda 执行交易的完整流程如下.

(1) 发送者创建交易并签名;

(2) 发送者发送交易数据及签名至接收者;

(3) 接收者验证交易数据及发送者签名无误,就附加上接收者签名;

(4) 接收者发送交易数据及交易双方签名至 Notary 共识服务;

(5) Notary 验证交易数据及交易双方签名无误,就附加上 Notary 签名;

(6) Notary 返回交易数据至接收者;

(7) 接收者核对 Notary 签名无误后提交交易;

(8) 接收者返回交易数据至发送者;

(9) 发送者核对接收者及 Notary 签名无误后提交交易.

Fig.3 Transaction flow in Corda

图 3 Corda 交易流程

在验证交易时,除了验证交易涉及的各方签名是否正确,交易双方主要验证交易数据是否符合智能合约中的约束条件,Notary 则主要检查交易是否涉及双花.图 3 的流程可以用 Corda 的 Flow 来定义,Flow 针对节点间的多轮交互,隐藏了网络、I/O 与并发等编程细节,支持用领域专用语言(domain specific language,简称 DSL)实现工作流编程.

2.3 Quorum交易流程

Quorum 扩展以太坊公有链,并加入了企业级区块链的特性.Quorum 交易分为公有交易和私有交易:公有交易是全网共享的传统以太坊交易,私有交易仅在交易双方共享.

Quorum 系统主要由 Quorum 节点和 Constellation 构成:Quorum 节点基于以太坊 Go 版本(go-ethereum)构建,主要用于执行合约及维护区块链与状态数据,状态数据分为存储公有交易执行结果的公有状态数据和存储私有交易执行结果的私有状态数据;Constellation 主要实现私有交易数据的加密、解密、存储及点对点传输.Quorum 公有交易的执行流程与以太坊类似,图 4 主要描述了私有交易的执行流程.

(1) 客户端发送私有交易到 Quorum 节点,并在交易中直接指明每个接收者的公钥;

(2) Quorum 节点将私有交易传送至对应的 Constellation 进行加密;

(3) Constellation 生成一个对称秘钥,先用该对称秘钥加密私有交易负载(payload),再分别用每个接收者的公钥分别加密该对称秘钥,最后还要基于私有交易的加密负载计算其哈希值;

(4) Constellation 将私有交易的加密负载、私有交易的加密负载的哈希值、加密后的对称秘钥分别点对点的传播至每个交易接收方的 Constellation;

(5) 数据传播成功后,Constellation 将私有交易的加密负载的哈希值返回至对应的 Quorum 节点;

(6) Quorum 节点将私有交易的加密负载的哈希值打包为一个以太坊交易,经以太坊 P2P 协议广播至所有 Quorum 节点;

(7) 该以太坊交易经过共识被打包进 Quorum 区块.当每个 Quorum 节点执行该以太坊交易时,需要基于该以太坊交易的负载(即私有交易加密负载的哈希值)向对应的 Constellation 请求原始的私有交易负载;

(8) 根据 Quorum 节点的请求,各个交易接收方的 Constellation 基于自己的私钥、公钥加密的对称秘钥、对称秘钥加密的私有交易负载解密出原始的私有交易负载;

(9) 交易接收方的 Constellation 将原始的私有交易负载返回至对应的 Quorum 节点.非交易接收方的 Constellation 没有接收过加密的私有交易负载,其向 Quorum 节点返回的是“NotARecipient”消息;

(10) 交易涉及的每个 Quorum 节点将私有交易提交至智能合约运行,智能合约会将执行私有交易时生成的状态数据存储至私有状态数据库.

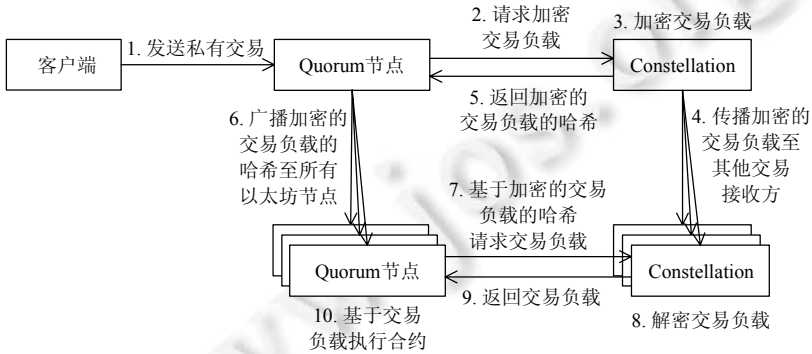

Fig.4 Transaction flow in Quorum

图 4 Quorum 交易流程

以太坊经历了全球范围应用考验且有强大的开发社区支持,基于以太坊构建的 Quorum 具有着先天的技术优势.Quorum 智能合约及 Dapp 开发与以太坊基本一致,因而以太坊中的智能合约和 Dapp 可轻松移植到 Quorum,Quorum 也可直接使用以太坊生态下的集成开发工具和开发框架.Quorum 以最小化修改 go-etnereum 并保留与以太坊的兼容性和互操作性为原则,以便及时跟随 go-ethereum 的版本变化,并促成以太坊最新技术在企业区块链中的应用.

3 区块链网络

企业级区块链必须提供节点准入机制,使得每个节点经过授权才可加入网络.因不适宜采用传统中心化的准入机制,目前主要依靠数字证书来识别每个节点,依靠数字签名来鉴别每次操作.公有链网络节点的身份是匿名的,因而需要在网络中将交易数据广播至所有节点,一方面保证在接收节点地址未知的情况下,交易数据仍能被传送到指定的接收方;另一方面,通过在每个节点接收并验证每笔交易,也阻止了双花的可能[48].在企业级区块链网络中,节点身份是已知的,并且需要控制数据的传播范围,所以其传播协议有着不同于公有链的设计.

3.1 准入机制

Fabric 节点中的 MSP(membership service provider)模块负责身份管理,主要完成数字证书验证、签名与验证、私钥管理等功能[49].Fabric 网络的各类节点(背书/提交节点、排序节点、客户端)由 X.509 数字证书表示其身份,也针对组织、管理员、普通用户生成相应的数字证书.普通用户一般发起与应用有关的商业交易,管理员则发起与系统相关的配置交易.一个组织代表一个机构,其下可包括背书/提交节点、管理员及普通用户.组织的证书是自签名的根证书,组织内的实体将该证书作为证书根.智能合约可依据调用者的数字证书、MSP ID 及其属性字段实现多种级别的访问控制.Hyperledger 还提供了独立的 Fabric-CA 项目(https://github.com/hyperledger/fabric-ca),其可作为 root CA 和 intermediate CA 为 Fabric 项目生成和撤销数字证书.

Corda 许可服务被称为 doorman[13],Corda 节点需要基于节点信息向 doorman 申请到根证书机构签名的 TLS 证书,才能加入到对应的 Corda 网络.为了控制交易数据的传播范围,交易发送者需在发送的消息中直接指定接收者地址.为了便于获取接收者地址,Corda 网络提供了包含节点地址、节点证书和节点服务等节点信息的网络地图(network map)服务[13].每个节点向网络地图服务上传签名的本节点信息,并周期地下载已签名的其他节点信息.任何节点都可提供网络地图服务,从而实现了网络地图服务的去中心化.

Quorum 扩展了以太坊 P2P 层,保证只有被授权的节点才可加入.Quorum 在每个节点都设置一个 JSON 配置文件,其定义了所有被授权的网络节点.如果要添加或移除节点,就需要修改所有节点的配置文件.为了实现节点的动态添加和移除,Quorum 计划基于智能合约实现节点准入机制.

3.2 网络协议

Gossip 协议的去中心化、可容错、最终一致性等特性非常适合于区块链网络,相对于全网广播协议,随机选取节点广播消息的 Goosip 协议减少了网络负载和攻击面.Fabric 网络采用 Gossip 协议保证了节点间传输消息的一致性,Gossip 协议在 Fabric 主要实现的任务是:

(1) 每个节点持续传播 alive 消息,以使系统及时发现出新的节点并监测离线节点;

(2) 为了减少与共识服务间的通信,每个组织选取一个主节点先从共识服务拉取区块数据,然后再将区块广播至组织内的其他节点;

(3) 缺失区块的节点以点对点方式从其他节点同步区块数据.

Fabric 的 Gossip 协议基于 gRPC 构建,并可利用 TLS 实现加密的数据传输.

为了保证交易数据仅对相关参与者可见,Corda 没有使用类似 Gossip 协议的广播通信,而是基于 AMQP1.0 协议实现了点对点的直接通信.AMQP1.0 协议报文是二进制的,相对于 REST 的文本格式报文,其传输效率更高.Corda 基于 AMQP1.0 协议并利用 TLS 实现了加密通信.

Quorum 采用 go-ethereum 的 P2P 传输层在 Quorum 节点间传播公有交易;Quorum 在私有交易中指明了接收者的公钥,所以 Constellation 直接将私有交易经 HTTPS 发送至接收者的 Constellation,没有参与私有交易的节点不会接收到交易.为了实现更高的区块传输效率,Quorum 采用 etcd Raft 的 HTTP 传输层传播区块数据.

4 共识机制

为了实现完全的去中心化,比特币、以太坊采用了 PoW 共识机制,虽然网络节点总数可以达到数以万计,

但是交易吞吐量非常低.另外,为了防止分叉带来的双花问题,交易提交一定时间后才可确认,但理论上所有交易并未得到最终确认[50],这显然无法适用于对交易结果有着严格确定性要求的企业级应用,尤其是金融行业应用.企业级区块链对交易吞吐量有着较高的要求,同时,其网络节点相对较少且规模基本稳定,因而更适合采用基于投票的 BFT(Byzantine fault-tolerant)[51,52]共识算法.企业级区块链节点身份通常是实名的,并且具有一定的可信性,为了提高整个系统的吞吐量,还可以采用仅容忍宕机错误的 CFT(crash fault-tolerant)[7,53]共识算法.

传统的区块链节点既需要执行共识协议又需要执行智能合约.Fabric 采用了合约执行、共识排序、验证写入相互解耦的系统架构,保证了各功能节点独立地进行扩展.因为共识服务不用执行交易和存储交易,即无需关心交易的具体内容,因而无状态的共识服务更易插件化[54].Fabric 提供了开发者使用的 Solo、高吞吐量的 Kafka 和基于 PBFT 的 BFT-SmaRt 这 3 种共识服务[55].

Corda 依靠 Notary 服务防止双花,避免产生冲突的交易.每笔交易提交前必须获得 Notary 服务的签名,以证明交易的每个输入状态所引用的资产都是未被花费的.Corda 提供了高信任场景下的单节点 Notary、基于 Raft 的 Copycat 和基于 PBFT 的 BFT-SmaRt 这 3 种类型的共识服务.Corda 网络可同时部署多种 Notray 服务,各个 Notary 服务可并行运行,用户可根据具体应用场景选择相应的服务.

Quorum 没有沿用以太坊的 PoW 共识机制.Quorum 同时维护着公有状态数据和私有状态数据:公有状态数据需要在全网所有节点间达成共识,私有状态数据仅需在交易参与者间达成共识.基于这些前提,Quorum 提供了基于 Raft 的 etcd 和基于 PBFT 的 Istanbul BFT 这两种共识服务.Quorum 也实现了基于 PoS 的 QuorumChain 共识服务,但在 Quorum2.0 中已被舍弃.

4.1 BFT-SMaRt

近年来,针对 BFT 的理论研究取得了重要进展,基于这些研究实现了一些原型系统,但很少被部署到真实系统中.业界成熟可用的产品相对较少,因为实现实用的 BFT 共识具有相当难度.里斯本大学基于 Java 的开源项目 BFT-SMaRt 则是一个较为成熟的系统,其基于 BFT 共识实现了 SMR(state machine replication),主要特性如下.

(1) 简洁性.着重协议正确性而避免限于琐碎的优化细节(如采用 Java 语言而非 C++);
(2) 模块化.基于模块化设计将系统划分为 SMR 模块(Mod-SMaRT)、状态同步(state transfer)、配置模块(reconfig)等;
(3) 可配置.既可配置 BFT 共识节点的动态加入和退出,又可配置采用 BFT 共识还是 CFT 共识;
(4) 可扩展.基于插件扩展系统功能;
(5) 多核感知.基于多核运行开销较高的计算任务(如签名验证);
(6) 高性能.在 4 个节点构成的局域网中,交易吞吐量达到了 80000TPS.

为了适应于广域网,Fabric 还参考了 WHEAT[56].WHEAT 是针对共识节点分布在广域网时的 BFT-SMaRT 改进版本,其主要在假设执行(tentative execution)和投票分配模式(voting assignment schemes)方面做了优化.假设执行是假设在 PBFT 共识的 prepare 阶段后系统不会出现异常,而在 prepare 阶段就提交请求,并让 commit 阶段异步执行;若 commit 阶段发生主节点更换,则回滚先前的执行结果.投票分配模式是给广域网中较快的共识节点分配较高的投票权重,以实现基于更少的节点更快的达成共识.

4.2 Notary

Corda 交易需要达成有效性共识和唯一性共识.

- 有效性共识需交易涉及的每个参与者都确认交易数据有效.为了达成有效性共识,既要检查交易是否符合每个输入状态和输出状态所引用的合约的约束条件,又要检查相应参与者的签名是否齐全.Corda 没有全局统一总账,每个节点一致性地存储着与自己业务相关的交易数据,所以为了确保发送者提供的交易是来源是可靠的,就需针对每个输入状态沿着 UTXO 模型一直回溯至最初的发行交易,以从其他节点获取到当前交易涉及的所有历史交易,并验证每笔交易都是有效的;
- 唯一性共识需 Notary 服务确认交易的每个输入状态所引用的输出状态都未被花费过,以阻止双花发

生.如果一笔交易的每个输入状态所引用的资产都未被花费,Notary 就会对该交易签名,并将相关信息记录在 Notary 内的一个 Map 中.Map 的 key 记录了上笔被花费交易的哈希与输出状态索引(*txID*,*outputIndex*),value 记录了当前被 Notary 签名的交易的哈希、输入状态索引及请求节点地址(*txID*,*inputIndex*,*requestingPeer*).Notary 基于 Map 中由其签名过的所有已花费交易,可快速验证一笔交易是否涉及双花.在检查一笔交易时,交易的所有输入状态必须指向同一 Notary 节点,否则就需将所有状态先迁移到同一 Notary 节点,其避免了在 Notary 服务中涉及两阶段提交.如果不涉及数据隐私,Notary 服务也可以运行有效性共识.由于 Notary 节点可以访问每笔交易的输入状态,在参与者互不信任且没有可信第三方的场景中,Notary 由哪方维护将会成为问题.

4.3 Istanbul BFT

以太坊支持的共识算法有基于 PoW 的 Ethash[57]和基于 PoA 的 Clique[58].为了在以太坊引入 PBFT 共识,AMIS(http://am.is)公司发布了基于 Go 语言的开源项目 Istanbul BFT.Istanbul BFT 根据 PBFT 实现了专门适用于以太坊的共识模块,其主要特性如下.

(1) 每个节点都会将其在 PBFT 共识收到的 2*f*+1 条 commit 签名消息写在区块头部的扩展字段,使得每个区块在共识上都是自验证的;
(2) 基于投票实现了共识节点的动态加入或退出;
(3) 利用 backlog 缓存失序的消息,避免了重传;
(4) 已接收到 2*f*+1 条 commit 消息的节点,即使未接收全 2*f*+1 条 prepare 消息,也可提前进入 commit 阶段;
(5) 所有节点间可以基于 Gossip 协议建立间接连接,而不需要建立一对一的直接连接.

5 区块链数据

区块链在数据结构上是以区块为结点的链表,每个区块中主要包含交易数据、前块哈希及元数据.区块中的交易数据是前一区块创建后到当前区块创建时,系统中发起的所有交易.针对每个区块都可计算出一个哈希值,其被称为区块哈希或区块 ID.通过在区块中包含前一区块的哈希,即可以前块哈希为指针,把各个孤立的区块链接起来.元数据主要包含了区块创建时的时间戳、块内交易数据哈希计算得出的 Merkle 根[59]等数据.以区块链组织交易,实现了交易数据的不可篡改性和可追溯性.为了支持智能合约的运行以及展现交易数据的执行结果,区块链系统通常提供了状态数据库.为了支持基于区块高度、区块哈希的区块数据检索及基于交易哈希的交易数据检索,区块链系统还提供了索引数据库.

5.1 Fabric区块链数据

Fabric 区块中的交易主要由读写集表示,读写集由背书节点依据交易数据执行智能合约后生成:读集表示执行该笔交易所需读出的数据集,每项读是一个健及其版本号(*key*,*ver*);写集表示存储交易执行结果所需写入的数据集,每项写是一个健及其新值(*key*,*val*).每次写入前,只有基于 MVCC(multiversion concurrency control)验证读集数据版本与状态数据库当前数据版本一致,才会将写集写入数据库.Fabric 实际上实现的是基于乐观锁的并发控制.为了方便新块的追加操作,Fabric 区块链数据以日志文件的方式进行存储.如图 5 所示: Fabric 除了支持状态数据库、区块索引库外,还提供了历史索引库,实现了基于主健的历史状态数据查询.Fabric 支持插件化的数据访问,底层数据库可选用 LevelDB 或 CouchDB.基于数据库产品自身特性,LevelDB 支持主健查询、复合主键查询、主健范围查询与主健历史查询.CouchDB 除了支持以上查询,还支持富查询及分页.

Fabric 区块链中除了包含交易数据的数据区块外,还有包含配置数据的配置区块,配置区块主要包含了区块链中所有节点的数字证书、共识服务地址、区块切分依据等系统配置参数.每条 Fabric 区块链的创世区块就是一个配置区块,如果需要增加 anchor 节点、调整区块尺寸等,就需要发送配置修改交易,经共识生成新的配置区块,再传播至其他节点以使其接受新的配置信息.

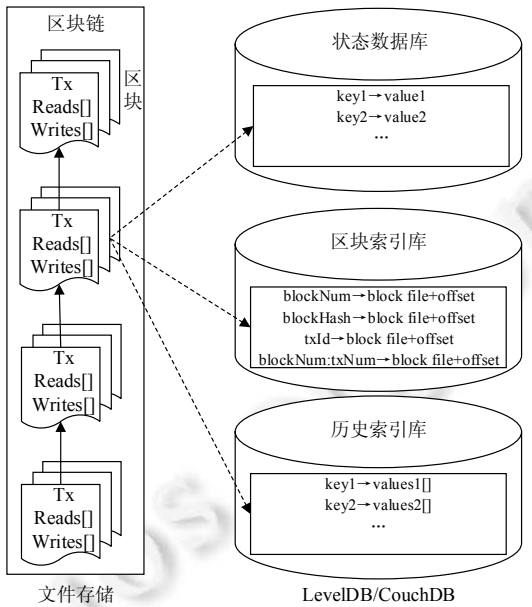

Fig.5 Blockchain data of Hyperledger Fabric

图 5 Hyperledger Fabric 区块链数据

5.2 Corda分布式账本

Corda 没有使用区块链组织交易数据.Corda 强调自己是受区块链系统启发的分布式账本,摒弃了区块链系统中不适合金融场景的设计元素.Corda 认为区块是 PoW 共识分摊挖矿开销的产物,以区块为单位提交交易会延迟交易的执行及增加端到端的延时,并不适合交易及时产生及时提交的金融业务场景.因为主要服务于数据可见范围严格受限的金融领域,Corda 系统没有维护一个全局账本,每个节点通过数据库一致性地维护与自己业务相关的当前和历史状态数据,该数据库被称为 Vault[13].Corda 采用 UTXO 模型,一个交易包含一到多个输入状态和一到多个输出状态,输入状态源自之前某笔交易的输出状态,执行交易就是花费之前交易中的一批输出状态,并生成一批新的输出状态.被花费的状态成为合约的历史状态,新的输出状态则成为了合约的当前状态.

如图 6 所示:交易 *n* 包含 3 个输入状态和 2 个输出状态,表示分别向 2 个账户转了 70 美元和 50 美元,其中有 80 美元来自于交易 *m*,另外的 10 美元和 30 美元分别来自于其他交易.每个输入状态通过交易哈希和输出状态索引(*txHash*,*outputIndex*)两个属性作为引用,指明自己花费的是哪笔交易中的哪个输出状态.输入状态按交易执行顺序把各个交易链接起来构成交易链,每笔交易可一直向前追溯至源头的原始发行交易(即交易不包含输入状态),向后可追踪至尚未花费的交易(即没有任何交易的输入状态指向该交易的输出状态).一个交易中被花费过的输出状态不能被再次花费,否则即被认为是双花,即至少有两笔交易的输入状态指向了同一交易的同一输出状态.与比特币的 UTXO 模型相比,Corda 的 UTXO 模型不但支持数字货币转账,还支持用户定义的通用数字资产的流转.Corda 交易还包含命令(command)、附件、时间窗口(time window)等内容,命令指明了交易类型(如转账、借贷),还提供了一个公钥列表,列出了需要对交易进行签名的各个交易者的公钥.附件是执行交易所需的一些数据文件,交易基于文件哈希来访问这些文件.时间窗口限定了交易执行的时间段.Corda 还提供了 Oracle[13]服务,其作为权威机构提供了交易所需的一些外部信息(如股价、汇率),Oracle 提供的外部信息主要被嵌在交易的命令或附件中.

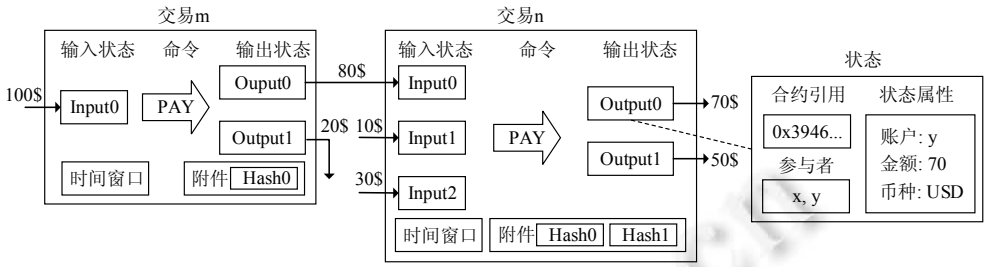

Fig.6 Transaction and state of Corda

图 6 Corda 的交易与状态

输入状态实质上是输出状态的引用,所以 Corda 中的状态通常是指输出状态.输出状态包含状态属性、合约引用和参与者列表.状态属性定义了执行交易时所涉及的各项状态数据,其可表达任意的业务对象(如股票、债券).合约引用定义了用于验证交易的智能合约的地址,参与者列表定义了该状态涉及的交易参与者.为了保留每笔交易的原始事实,Corda 中的状态对象是不可变的(immutable),以反映合约各个阶段的真实状态.执行交易并没有直接修改账本中的状态数据,实际上是将该状态标记成为历史状态(即已花费),并创建新的状态以反映修改后的当前状态,所有历史状态与当前状态构成的序列展现了完整的业务事实,这非常适合于注重原始单据合法合规性的金融行业.在基于账户的模型中,对同一账户的所有修改操作都要串行执行; UTXO 模型中的每个输出状态是独立且不可变的,花费一个输出状态不会影响其他输出状态,因而多笔交易可并行执行.

Corda 底层采用通用关系数据库(缺省是 H2 数据库)存储交易数据和状态数据,可实现 Corda 账本与企业内部信息系统在数据库层面上的无缝整合,避免了跨系统汇总查询引起的数据迁移、核对及同步.Corda 支持基于 JPA(Java persistence architecture)的数据持久化,可实现复杂的 SQL 查询及与链下数据的连接查询.传统区块链系统节点丢失数据时,主要从其他对等节点同步缺失的区块来恢复数据,但 Corda 主要利用底层关系数据库的容灾备份机制来恢复丢失的数据.

5.3 Quorum区块链数据

以太坊所有交易数据存储在区块链上,每个交易数据的执行结果存储在状态数据库,区块链与状态数据库都构建在 LevelDB 数据库上,任何人都可以访问所有交易数据和状态数据,以太坊数据的全网可见性显然无法满足企业级区块链的实际需求.因此,Quorum 将系统中的交易分为公有交易和私有交易,以对其采用不同的交易流程和存储.相对于公有交易,私有交易加入了一个可选参数 privateFor,它包含了多个接收者公钥的列表,指明了该私有交易应该只发送给那些接收者.图 7 描述了 Quorum 区块链数据的组织方式.

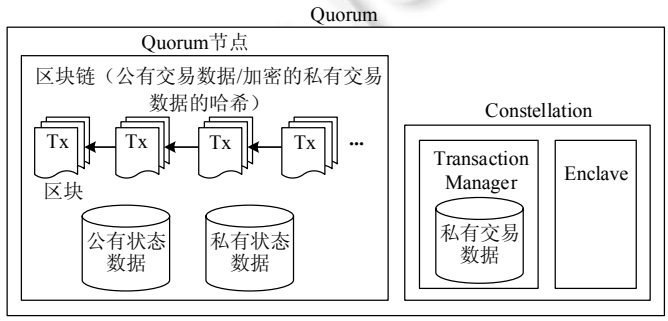

Fig.7 Blockchain data of Quorum

图 7 Quorum 区块链数据

Quorum区块中存储的是公有交易以及私有交易负载被加密后的哈希值,其既保护了私有交易隐私,又实现了私有交易和公有交易参与统一的共识.私有交易数据存储在链外(off-chain),也就是 Constellation.私有交易在智能合约中的执行结果存储在 Quorum 节点的私有状态数据库中.Quorum 节点包括公有状态数据和私有状态数据,所以其需要同时维护公有状态 Merkle Patricia 树和私有状态 Merkle Patricia 树.所有节点都可访问的数据是公有状态数据和交易数据哈希,所以 Quorum 区块的验证工作主要检验公有状态 Merkle Patricia 树根和交易 Merkle Patricia 树根.

6 智能合约

区块链在互不信任的多方间实现了数据的可信共享,运行在区块链上的智能合约则在互不信任的多方间实现了商业逻辑的可信执行.智能合约是运行在区块链上的程序代码,同时执行在多个区块链节点上,任何参与者都无法强行将其停止.智能合约定义了交易规则,外部应用通过调用智能合约来执行各种交易及访问区块链数据.交易数据被记录在区块链,合约执行结果则被记录在状态数据库.

6.1 合约模型

Fabric 智能合约被称为 Chaincode,其主要用于执行交易和访问状态数据.编写合约就是实现 Chaincode 接口中的 *Init* 和 *Invoke* 函数,以执行状态初始化和读写状态数据.Chaincode 运行在背书节点,但不同于传统区块链,Chaincode 无需在所有的背书节点上运行.在部署 Chaincode 时,可依据背书策略让 Chaincode 运行在部分指定的背书节点.背书策略定义了执行 Chaincode 所需的背书节点数量及组合(如一个 Chaincode 至少要被 n 个背书节点中的任意 k 个节点执行并签名),用户可根据不同应用所需的信任模型,灵活地定义背书策略.在可信环境下,执行 Chaincode 背书节点的数目越少,系统资源会被占用得越少,也会越快收集全执行结果;空闲的背书节点可同时执行其他 Chaincode,从而实现了 Fabric 智能合约的并行执行.另外,指定 Chaincode 仅在可信节点上部署运行,可避免合约逻辑与交易数据在不可信节点上的传播与泄露.

Corda 交易包含多个输入/输出状态,每个状态都有一个引用指向一个智能合约,执行交易时,需要执行交易中每个状态所引用的合约.为了支持金融行业严格的审计需求,Corda 智能合约主要验证交易数据,以保证其符合各项约束条件.编写 Corda 合约就是实现 Contract 接口中的 *verify* 函数,*verify* 函数以交易数据为输入,定义了具体的验证规则,如果不符合规则就抛出异常.Corda 的合约是无状态的,其不负责存储任何数据,而由 Flow 在共识达成后才将数据存储在交易双方的节点.为了保证业务合法合规,Corda 智能合约有一个哈希引用指向了该合约所依据的原始法律文档,以作为处理纠纷时的法律依据.

Quorum 智能合约分为公有合约和私有合约,它们在编程实现上并无分别:公有合约运行在所有节点,以公有交易数据为输入,将执行结果存储在公有状态数据库;私有合约仅运行在与交易相关的参与者节点,以私有交易数据为输入,将执行结果存储在私有状态数据库.因为私有状态数据的访问限制,Quorum 公有合约不能调用私有合约.私有合约可以调用公有合约,但仅允许执行读操作,不能执行写操作.Quorum 沿用了以太坊的智能合约模型,但以太坊执行合约按 Gas[60]计费的设计显然并不适用于企业级区块链.为了避免长时间运行的合约阻塞整个系统,Quorum 平台执行合约仍然消耗 Gas,只是将 Gas 的价格(gasPrice)设置为 0,从而绕过了 Gas 计费的问题.

6.2 沙箱环境

智能合约不能直接运行在区块链节点上.因为一方面要保证在不同节点的软硬件环境下,合约执行结果始终是确定的;另一方面,合约中若含有漏洞或恶意代码,要保证不会影响其他合约的执行及区块链节点的安全.所以智能合约必须运行在沙箱环境中.目前,沙箱主要分为虚拟机和容器两类,都是为了保证合约代码在沙箱中执行时,对合约使用的资源进行限制和隔离.

Fabric 使用轻量级的 Docker 容器作为沙箱,其基于 Docker 的隔离性和安全性,保护了宿主机不受容器内恶意合约的攻击,也防止了容器之间的相互影响.Chaincode 可基于 Go,Node.js 和 Java 开发,这些语言不但图灵完

备,编译技术成熟,也减轻了合约编程者的学习门槛.Fabric 智能合约的执行结果可能是不确定的,因为其没有限制使用编程语言中一些非确定性的特性.Fabric 依据背书策略收集并比较多个背书节点的合约执行结果,从而避免了不一致数据的提交.

为了保证合约执行结果的确定性,Corda 采用一个修订版的 JVM 作为沙箱,限制引用编程语言中一些非确定性特性.同时,对合约生成的字节码做了静态分析和重写,限制合约运行时间过长及使用内存过多.Corda 合约可基于 Kotlin 和 Java 开发,并基于 Kotlin 和 Java 定义了领域专用语言,使得智能合约编写更为简洁和高效,也提高了合约代码的可读性.

Quorum 沿用了以太坊自定义的 EVM 作为沙箱,运行以太坊自定义的字节码,这些字节码不能访问 EVM 宿主机的网络系统、文件系统和其他进程,合约之间也只有有限的调用.Quorum 合约需基于以太坊自定义的 Solidity 语言开发,之所以创建新的开发语言,就是为了保证合约执行结果的确定性.Solidity 虽是一个执行结果完全确定的编程语言,但对企业应用开发者而言,学习新的语言需要一个过程,且 Solidity 对复杂数据结构支持有限,各种功能的函数库也并不完备,因而实现智能合约受到一定的限制.

7 隐私保护

比特币、以太坊的每个节点全量存储着全部交易数据,每个用户可见证任何用户的交易历史.这种全网存储与全网见证虽保证了数据的可靠性,但却牺牲了数据隐私性.企业级区块链的用户主要是商业机构,交易数据是其重要资产和商业机密,隐私保护是必须具备的特性.

7.1 通 道

为了确保数据的隐私性,Fabric 提出了通道(channel)[61]的方案,图 2 中的交易流程就是在一个通道内实现的.多个参与者可自发组建一个通道,每个通道拥有一条专有的区块链,通道内的所有交易数据存储在内部的链上,只有通道内的节点才可访问链上的数据,没有加入通道的节点将无权访问.Fabric 通过将交易分配到相互隔离的多条区块链上,实现了私密的交易,保障了数据的隐私性.通道实际上是一种基于多链的数据分区,一个节点根据交易需求可加入多条通道,多个通道内的交易可并行地执行.相对于单链的方案,其提高了全网的交易吞吐量.如果有 n 个交易者,每个交易者都需和其他 n–1 个交易者进行仅涉及双方的私密交易,那么 Fabric 就需建立 $n(n-1)/2$ 个通道及区块链,此时,通道的开销就会给系统造成很大压力.Fabric 智能合约目前仅支持跨通道的查询操作,实现跨通道的修改操作需要在业务层实现.此外,如果交易者需要统计所有通道中的交易数据,也会相当繁琐.

7.2 Flow

Corda 基于 Flow 用代码定义了多个参与者间的交易流程,实现了无中心节点控制的交易流程自动化.Flow 中的交易仅在交易参与者间共享与执行,交易数据被点对点地直接发送到指定的接收者.一笔交易在流程中需在多个节点间进行多轮通信,每个节点还需进行验证和签名,用传统编程方法实现整个流程会比较困难,而 Flow 封装了网络与签名等编程细节,提供了简洁的工作流编程接口.Flow 是基于纤程(fiber)实现的,当等待其他节点的消息时,Flow 的当前堆栈会被挂起并被序列化到底层的数据库,以释放占有的系统资源;当其他节点的消息到达时,被挂起的 Flow 会被唤醒并恢复堆栈,从而可从中断处继续执行.阻塞的 Flow 并不占用内存资源,因而可实现更高的 Flow 并发度.Flow 的纤程基于 Quasar 库构建,Flow 代码实质上是一个运行在纤程中的异步可持久化状态机.多条 Flow 可并行执行,每个参与者也可同时执行多个 Flow,从而提高了系统的交易吞吐量.

为了实现完善的隐私保护,Corda 还采用了其他技术:首先,在每次交易中使用随机公钥隐藏了交易双方的真实身份;其次,为了减少交易数据的泄露,在对交易进行签名时,Corda 利用 Tear-offs 技术实现了对签名者仅展示交易的部分数据(例如,Notary 服务仅能访问交易的输入状态,Oracle 服务仅能访问交易的命令部分),而不展示交易的全部数据.Tear-offs 将一个交易的各组成部分的哈希值作为 Merkle 树的叶子结点,并依照 Merkle 树定义计算出 Merkle 根(即交易 ID),当 Notary 需要对交易签名时,可仅展示输入状态及其到根结点路径上的各直接

分支结点,从而隐藏该交易的其他部分内容(如输出状态、命令和时间窗口等).

7.3 Constellation

为了实现隐私保护,Quorum 平台针对公有交易和私有交易分别使用了不同的交易流程和状态数据库,其中,私有交易的加解密、存储及点对点传输都是依靠 Constellation 来实现的.如图 7 所示,Constellation 主要包含交易管理(transaction manager)和 Enclave 两个模块.

- 交易管理主要实现了私有交易的链外存储、基于加密负载哈希的私有交易负载访问、点对点的传输私有交易的加密负载至指定接收者;
- Enclave 负责私有交易负载的加解密,其加密私有交易负载的流程为:(1) 接收交易管理发送的私有交易并验证交易发送者的签名;(2) 生成对称秘钥;(3) 使用对称秘钥加密私有交易负载;(4) 计算私有交易加密负载的哈希;(5) 逐一使用每个交易接收者的公钥加密对称秘钥;(6) 返回加密的私有交易负载、加密的私有交易负载的哈希、被加密的对称秘钥到交易管理.解密私有交易负载的流程与之相反,流程中使用对称秘钥是为了解决非对称秘钥加解密效率低的问题.Enclave 还负责管理私钥,实际上是一个与其他模块隔离的虚拟 HSM.

通过依靠 Constellation 进行私有交易数据管理,没有参与私有交易的节点不接收私有交易、不存储私有交易、不执行私有交易,其仅能访问到私有交易负载被加密后的哈希值.

8 研究挑战与趋势

企业级区块链引起了以金融行业为主的众多企业机构的关注.当应用于实际业务时,企业级区块链目前在诸多方面尚存在问题.为了解决这些问题,未来需要在以下几个方面进行更深入的研究.

(1) 吞吐量

目前,依据各方独立提供的数据,Fabric 的交易吞吐量约为 3500TPS[43],Corda 的交易提供吞吐量约为 1000TPS[62],Quorum 的交易吞吐量约为 100TPS[16],与比特币(7TPS)、以太坊(15TPS)相比已有相当改善,但与传统的数据库系统相比,仍有着不小的差距.区块链系统虽不涉及复杂的事务处理,但是其以区块为单位提交交易、每笔交易涉及的多方签名与验证、基于 BFT 的共识机制、串行执行的智能合约等都会限制其吞吐量.另外,在实际部署应用时,不同的策略与配置也会影响吞吐量.以 Fabric 为例,并行运行的通道数目、通道内并行运行的背书节点数目、区块内多笔交易修改同一状态的写冲突程度、底层数据库用 LevelDB 还是 CouchDB、每个区块包含的交易数目(缺省为 10)及出块间隔(缺省为 2s)都会影响系统吞吐量.因此,无论从系统架构设计还是从实际部署维护,企业级区块链在吞吐量方面尚需更多的研究.

(2) 共识机制

首先,作为区块链系统的核心引擎,共识机制目前仍是整个系统性能的关键瓶颈.当前,许多区块链平台声称提出了高性能的共识机制,但这些共识并没给出前提假设、数学模型和形式化证明,其安全性无法衡量.能够在安全与性能间取得最佳权衡的共识机制,仍是最迫切的研究工作.其次,企业级区块链平台未必要自己实现共识机制,完全可以采用通用的解决方案,如 Fabric 和 Corda 采用了 BFT-SMaRt,Burrow 和 Cosmos 采用了 Tendermint.因此,通用的 BFT 与 CFT 共识框架将会是未来的一个研究方向.最后,因为没有 one-size-fits-all 的共识机制[63],同一共识在不同的安全假设与可信场景下会展现出不同的性能.所以企业级区块链平台必须支持插件化、可配置的共识服务,以灵活支持从局域网到广域网、从 BFT 到 CFT 不同场景下的多种共识.

(3) 可扩展性

比特币、以太坊采用的是全网共享一条区块链的单链方案,每个节点需处理、存储全网的所有交易,整个区块链系统能力实际上受限于单个节点的处理能力.企业级区块链的每个节点通常仅处理、存储与自己业务相关的交易数据,使其更易于支持可扩展性[64].为了实现可扩展性,Fabric 采用了多通道(multichannel)的数据分区方案,使得多通道(即多链)间可独立并行地处理交易.在同一通道内,Fabric 多个合约可并行运行在不同的背书节点,通过增加背书节点,就可增加并行运行的智能合约数目.Corda 可通过增加节点数来提高整个网络的 Flow

并发度,其也支持多个 Notray 服务集群的并行运行.从整体上来讲,与通过增加节点数而线性提高系统吞吐量和容量的横向扩展性目标相比,企业级区块链尚有很大的差距.因此,如何更高效灵活地实现可扩展性,将是研究的热点.

(4) 隐私保护

区块链隐私保护的难点在于既需隐藏交易细节,又需验证交易的有效性.目前,Fabric,Corda 与 Quorum 提供的隐私保护方案都有着一定的局限性:Fabric 通道面临着仅两个交易者也需建立通道所带来的开销问题以及排序服务可读取所有通道交易数据的问题;Corda 处理每笔交易前都需验证从当前交易到发行交易之间的全部历史交易,存在着交易验证流程低效及历史交易数据泄露的问题;Quorum 私有状态数据无法被非交易者节点读取见证,因而无法解决跨私有合约交易时的双花问题.因此,各平台都在积极研究更为完善的隐私保护方案.Fabric 为了进一步在通道内实现细粒度的隐私保护,其 PDC(private data collection)方案将状态数据库分为公有状态数据库和私有状态数据库(SideDB),私有交易数据仅在通道内部分指定的节点间传输,排序服务处理的是私有交易的读写集的哈希值,区块链存储的也是私有交易的读写集的哈希值,私有交易的写集被写入到了隔离的私有状态数据库,公有状态数据库存储的是私有状态数据健值的哈希.Intel SGX 基于 CPU 提供了可信内存空间 enclave,任何恶意软件甚至操作系统都无法访问和影响 enclave 内部的敏感代码和数据.Corda 采用 Intel SGX 技术实现了加密交易在 enclave 内的解密与验证,使得处理每笔交易时的历史交易验证都在 enclave 内完成,从而避免了历史交易数据的泄露.零知识证明(zero-knowledge proof)可以让证明者在不透漏任何有用信息的前提下,使验证者相信某个论断的正确性.Quorum 基于非交互的零知识证明 zk-SNARKs[65,66]构建了零知识安全层(zero knowledge security layer)[67],在不泄露发送方、接收方和交易金额的前提下,基于交易数据生成的零知识证明,使得全网均可验证交易的有效性.为了提供一个将企业级区块链和可信计算环境 TEE(trusted execution environment)高度集成的基础平台,微软提出了 Coco 框架(confidential consortium blockchain framework)[68],其保证了运行于 TEE 内的区块链代码和数据的机密性和完整性,提高了部署在 Coco 框架上的区块链平台的隐私性.目前,与其合作的企业级区块链平台有 Corda,Quorum 与 Hyperledger Sawtooth.此外,基于同态映射实现加密数据运算的同态加密(homomorphic encryption)[69]和实现智能合约中私密信息处理的 Hawk[70]也都引起了业界的关注.

(5) 跨 链

如同 Intranet 网络之间需要互联互通,企业级区块链网络间也需要互联互通.目前较有影响力的跨链技术是 Polkadot[71],Cosmos[72]和 Hyperledger Quilt(https://github.com/hyperledger/quilt).Polkadot 的主干网络被称为中继链(relay chain),其以以太坊为主实现了与各种平行链(parachain)的互联,每个平行链就是一个单独的区块链网络.Polkadot 还以其他公有链为升级目标,最终让以太坊可直接与任何链进行互联.Cosmos 把不同种类的区块链子网看做 Zone,通过主干网络 Cosmos Hub 上运行的 IBC(inter-blockchain communication)协议,实现不同 Zone 之间的互联.Hyperledger Quilt 是 ILP 协议的 Java 实现,ILP 是无需中介的跨账本支付协议,其实现了不同数字资产账本间的自动路由与原子支付操作.Polkadot 专注于实现通用的跨链通信,Cosmos 专注于实现跨链的数字资产交易,Hyperledger Quilt 则专注于实现跨链的支付操作.各类企业级区块链平台无论在系统整体架构还是在交易数据格式上都存在着很大差别,这些都加大了跨链技术的研究难度.

(6) 评测系统

基准评测系统可帮助使用者对比各区块链平台,以选择适合自己需求的平台;也可帮助系统设计者识别出系统缺陷以进行改进.Dinh 等人[73]开发了专用于评测企业级区块链的开源评测框架 Blockbench(https://github.com/ooibc88/blockbench),其将区块链平台分为共识层、数据模型层、智能合约执行层和应用层,提供了整体性能评估的宏观评测基准和分层性能评估的微观评测基准,具体可从吞吐量、延迟、可扩展性、容错性和安全性这 5 个维度进行评测,目前支持对 Fabric,Ethereum 和 Parity(https://www.parity.io)的评测.Hyperledger Caliper (https://github.com/hyperledger/caliper)是主要由华为开发的区块链开源测评工具,其通过可插拔的适配层来集成不同区块链平台,目前支持对 Fabric,Sawtooth 和 Iroha 的评测.Hyperledger Caliper 基于一组预定义用例,从吞

吐量、延迟和资源利用率这 3 个维度进行评测.在评测区块链平台多个维度的指标中,以交易吞吐量为代表的性能评测最受关注.对不同区块链平台而言,网络节点数目不同、共识机制容错类型不同、交易模型及尺寸不同、智能合约功能复杂度不同、测试用例不同等,都直接影响着其所能展示出来的性能.这就要求评测系统不仅要基于多个维度指标进行综合评估,还要对区块链平台内在的系统架构、交易流程、应用领域等方面有着深入的理解.为了实现更为客观、公平的评测,也非常需要制定可被行业所广泛接受的通用评测基准.

(7) 底层数据库

不同于以数字货币业务为主的公有链,企业级应用通常涉及复杂的统计查询与分析.由表 3 可知,目前,企业级区块链底层数据库大多是 NoSQL 数据库,其主要支持的是 Key-Value 查询.但现有的技术人员更熟悉 SQL 查询,现有的数据分析工具更多基于 SQL 构建,如同支持 Hadoop 的 SQL 查询工具 Hive、支持 HBase 的 SQL 查询工具 Phoenix,目前迫切需要支持企业级区块链的 SQL 查询工具.比特币、以太坊等公有链底层都采用了 LevelDB 单机数据库,这是因为轻量级 LevelDB 数据库所需配置较低,可运行在更多常规计算能力的节点上,从而更易实现公有链去中心化的理念.节点数目有限的企业级区块链事实上是弱中心化的,其更需要支持高吞吐、高并发、丰富查询、权限控制、备份与恢复等特性的适合于企业级应用的数据库.

(8) 智能合约

首先,相对于公有链,企业级区块链中的业务逻辑更为复杂,需要智能合约支持更复杂的功能与数据结构,同时还要确保执行结果的确定性以及合约语言的易用性,这对智能合约语言的设计提出了更高的要求;其次,The DAO 合约漏洞、Parity 多重签名合约漏洞等揭示了智能合约的大量安全问题[74],相对于传统的软件测试、专家审查等方法,研究智能合约的语义模型,并通过形式化验证检测合约漏洞[75]以保障智能合约的运行时安全,将是今后的主要研究方向;最后,智能合约发布后,因需求变更、Bug 修复等原因,如何实现不停机的全网合约同步升级以及向下兼容先前版本的状态数据,也是必须解决的问题.

(9) 可治理性

在部署实施企业级区块链平台时,区块链的去中心化给目前的平台带来了许多治理上的问题:首先,在节点与用户管理方面,传统中心化的数据库基于用户名与密码验证的实现已非常成熟,而去中心化的区块链基于公钥与私钥验证的实现仍需大量繁琐的手工配制,私钥的存储与丢失问题也需要易用可行的解决方案;其次,去中心化的区块链系统没有控制全局的管理员,系统在软件动态升级、节点动态加入等方面需要联盟成员集体表决并签名,其造成了决策的低效与滞后;最后,区块链的去中心化与政府部门与监管机构的监管政策相矛盾,如何在实现去中心化同时兼顾政府与监管的合规性要求,也是需要研究的问题.

9 结束语

关系数据库基于关系模型、事务处理、查询优化等技术,解决了以银行为代表的金融机构内部记账的业务需求.区块链技术通过集成 P2P 协议、块链结构、共识机制、智能合约等技术,解决了多个互不信任机构间一致性记账的业务需求,减少了跨机构业务流程中的摩擦,缩短了对账周期、降低了运营成本、提升了协作效率.目前,Hyperledger Fabric,Corda 和 Quorum 等企业级区块链平台在以金融机构为代表的众多企业应用实施时,还主要以概念验证(proof of concept,简称 POC)为主.与发展了近 40 年的传统数据库相比,企业级区块链技术仍处于技术路线的初期阶段,从系统架构到开发范式都尚未形成统一标准,在诸多方面还存在许多问题.为了解决这些问题,未来的还需更多的研究工作.

References:

[1] Swan M. Blockchain: Blueprint for a New Economy. Sebastopol: O'Reilly Media, Inc., 2015.

[2] Tapscott D, Tapscott A. Blockchain Revolution: How the Technology Behind Bitcoin and Other Cryptocurrencies is Changing the World. New York: Portfolio, 2016.

[3] Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. White Paper, 2008.

[4] Buterin V. A Next-Generation Smart Contract and Decentralized Application Platform. White Paper, 2014.

[5] King S, Nadal S. PPCoin: Peer-to-peer crypto-currency with proof-of-stake. 2012. http://peerco.in/assets/paper/peercoin-paper.pdf

[6] Castro M, Liskov B. Practical byzantine fault tolerance and proactive recovery. ACM Trans. on Computer Systems, 2002,20(4): 398−461. [doi: 10.1145/571637.571640]

[7] Ongaro D, Ousterhout JK. In search of an understandable consensus algorithm. In: Proc. of the 2014 USENIX Annual Technical Conf. (ATC). 2014. 305−319.

[8] Anati I, Gueron S, Johnson S, Scarlata V. Innovative technology for CPU based attestation and sealing. In: Proc. of the 2nd Int'l Workshop on Hardware and Architectural Support for Security and Arivacy. 2013. 13.

[9] Olson K, Bowman M, Mitchell J, Amundson S, Middleton D, Montgomery C. Sawtooth: An Introduction. White Paper, 2018.

[10] Van Reesse R, Schneider FB. Chain replication for supporting high throughput and availability. In: Proc. of the 6th Symp. on Operating System Design and Implementation (OSDI). 2004. 91−104.

[11] Kwon J. Tendermint: Consensus without mining. Technical Report, 2014.

[12] Aublin PL, Mokhtar SB, Quéma V. RBFT: Redundant byzantine fault tolerance. In: Proc. of the 33rd Int'l Conf. on Distributed Computing Systems (ICDCS). 2013. 297−306. [doi: 10.1109/ICDCS.2013.53]

[13] Hearn M. Corda: A Distributed Ledger. White Paper, 2016.

[14] Brown RG, Carlyle J, Grigg I, Hearn M. Corda: An Introduction. White Paper, 2016.

[15] Walport M. Distributed ledger technology: Beyond block chain. Technical Report, 2016.

[16] JPMorgan Chase & Co. Quorum Whitepaper. White Paper, 2016.

[17] Greenspan G, MultiChain Private Blockchain. White Paper, 2016.

[18] Schwartz D, Youngs N, Britto A. The Ripple Protocol Consensus Algorithm. White Paper, 2014.

[19] Thomas S, Schwartz E. A protocol for interledger payments. 2015. https://interledger.org/interledger.pdf

[20] McConaghy T, Marques R, Müller A, De Jonghe D, McConaghy T, McMullen G, Henderson R, Bellemare S, Granzotto A. BigchainDB: A Scalable Blockchain Database. White Paper, 2016.

[21] Yuan Y, Wang FY. Blockchain: The state of the art and future trends. Acta Automatica Sinica, 2016,42(4):481−494 (in Chinese with English abstract). [doi: 10.16383/j.aas.2016.c160158]

[22] He P, Yu G, Zhang YF, Bao YB. Survey on blockchain technology and its application prospect. Computer Science, 2017,44(4):1−7 (in Chinese with English abstract). [doi: 10.11896/j.issn.1002-137X.2017.04.001]

[23] Tschorsch F, Scheuermann B. Bitcoin and beyond: A technical survey on decentralized digital currencies. IEEE Communications Surveys and Tutorials, 2016,18(3):2084−2123. [doi: 10.1109/COMST.2016.2535718]

[24] Dinh TTA, Liu R, Zhang M, Chen G, Ooi BC, Wang J. Untangling blockchain: A data processing view of blockchain systems. IEEE Trans. on Knowledge and Data Engineering, 2018,30(7):1366−1385. [doi: 10.1109/TKDE.2017.2781227]

[25] Qian WN, Shao QF, Zhu YC, Jin CQ, Zhou AY. Research problems and methods in blockchain and trusted data management. Ruan Jian Xue Bao/Journal of Software, 2018,29(1):150−159 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5434.htm [doi: 10.13328/j.cnki.jos.005434]

[26] Shao QF, Jin CQ, Zhang Z, Qian WN, Zhou AY. Blockchain: Architecture and research progress. Chinese Journal of Comptuers, 2018,41(5):969−988 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2018.00969]

[27] Donet JAD, Pérez-Sola C, Herrera-Joancomartí J. The bitcoin P2P network. In: Proc. of the Financial Cryptography and Data Security. 2014. 87−102. [doi: 10.1007/978-3-662-44774-1_7]

[28] Demers A, Greene D, Hauser C, Irish W, Larson J, Shenker S, Sturgis H, Swinehart D, Terry D. Epidemic algorithms for replicated database maintenance. In: Proc. of the 6th Annual ACM Symp. on Principles of Distributed Computing (PODC). 1987. 1−12. [doi: 10.1145/41840.41841]

[29] Karp R, Schindelhauer C, Shenker S, Vöcking B. Randomized rumor spreading. In: Proc. of the 41st Annual Symp. on Foundations of Computer Science (FOCS). 2000. 565−574. [doi: 10.1109/SFCS.2000.892324]

[30] OASIS. Advanced message queuing protocol (amqp) version 1.0. Technical Report, 2012.

[31] Antonopoulos AM. Mastering Bitcoin: Unlocking Digital Cryptocurrencies. Sebastopol: O'Reilly Media, Inc., 2014.

[32] Duan S, Meling H, Peisert S, Zhang H. BChain: Byzantine replication with high throughput and embedded reconfiguration. In: Proc. of the 18th Int'l Conf. on Principles of Distributed Systems (OPODIS). 2014. 91−106. [doi: 10.1007/978-3-319-14472-6_7]

[33] Narkhede N, Shapira G, Palino T. Kafka: The Definitive Guide. Sebastopol: O'Reilly Media, Inc., 2017.

[34] Bessani A, Sousa J, Alchieri E. State machine replication for the masses with BFT-SMaRt. In: Proc. of the 44th Annual IEEE/IFIP Int'l Conf. on Dependable Systems and Networks (DSN). 2014. 355−362. [doi: 10.1109/DSN.2014.43]

[35] Saito K, Yamada H. What's so different about blockchain? Blockchain is a probabilistic state machine. In: Proc. of the 36th IEEE Int'l Conf. on Distributed Computing Systems Workshops. 2016. 168−175. [doi: 10.1109/ICDCSW.2016.28]

[36] Narayanan A, Bonneau J, Felten E, Miller A, Goldfeder S. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton: Princeton University Press, 2016.

[37] Anderson JC, Lehnardt J, Slater N. CouchDB: The Definitive Guide. Sebastopol: O'Reilly Media, Inc., 2010.

[38] Popov S. The Tangle. White Paper, 2016.

[39] Baird L. The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance. White Paper, 2016.

[40] Szabo N. Smart contracts: Building blocks for digital markets. 1996. http://www.alamut.com/subj/economics/nick_szabo/smart Contracts.html

[41] Yang BH, Chen C. Principle, Programming and Applications of Blockchain. Beijing: China Machine Press, 2017 (in Chinese).

[42] Dannen C. Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners. Berkeley: Apress, 2017.

[43] Androulaki E, Barger A, Bortnikov V, Cachin C, Christidis K, Caro AD, Enyeart D, Ferris C, Laventman G, Manevich Y, Muralidharan S, Murthy C, Nguyen B, Sethi M, Singh G, Smith K, Sorniotti A, Stathakopoulou C, Vukolic M, Cocco SW, Yellick J. Hyperledger fabric: A distributed operating system for permissioned blockchains. In: Proc. of the 13th EuroSys Conf. (EuroSys), Vol.30. 2018. 1−15. [doi: 10.1145/3190508.3190538]

[44] Schneider FB. Implementing fault-tolerant services using the state machine approach: A tutorial. ACM Computing Surveys, 1990, 22(4):299−319. [doi: 10.1145/98163.98167]

[45] Cachin C, Schubert S, Vukolić M. Non-Determinism in Byzantine fault-tolerant replication. In: Proc. of the 20th Int'l Conf. on Principles of Distributed Systems (OPODIS), Vol.24. 2016. 1−16. [doi: 10.4230/LIPIcs.OPODIS.2016.24]

[46] Budhiraja N, Marzullo K, Schneider FB, Toueg S. Distributed Systems. 2nd ed., New York: ACM Press/Addison-Wesley, 1993. 199−216.

[47] Kapritsos M, Wang Y, Quema V, Clement A, Alvisi L, Dahlin M. All about Eve: Execute-verify replication for multi-core servers. In: Proc. of the 10th Symp. on Operating Systems Design and Implementation (OSDI). 2012. 237−250.

[48] Garay JA, Kiayias A, Leonardos N. The Bitcoin backbone protocol: Analysis and applications. In: Proc. of the 34th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Vol.2. 2015. 281−310. [doi: 10.1007/978-3-662-46803-6_10]

[49] Zhang ZJ, Dong N, Zhu XT, Chen JX. Inside the Blockchain: Technology and Application of Hyperledger. Beijing: China Machine Press, 2018 (in Chinese).

[50] Vukolić M. The quest for scalable blockchain Fabric: Proof-of-work vs. BFT replication. In: Proc. of the Open Problems in Network Security. 2015. 112−125. [doi: 10.1007/978-3-319-39028-4_9]

[51] Lamport L, Shostak R, Pease M. The Byzantine generals problem. ACM Trans. on Programming Languages and Systems, 1982, 4(3):382−401. [doi: 10.1145/357172.357176]

[52] Cachin C, Vukolić M. Blockchains consensus protocols in the wild. In: Proc. of the 31th Symp. on Distributed Computing. 2017. 1−16.

[53] Junqueira FP, Reed BC, Serafini M. Zab: High-performance broadcast for primary-backup systems. In: Proc. of the 2011 IEEE/IFIP Int'l Conf. on Dependable Systems and Networks (DSN). 2011. 245−256. [doi: 10.1109/DSN.2011.5958223]

[54] Yin J, Martin JP, Venkataramani A, Alvisi L, Dahlin M. Separating agreement from execution for Byzantine fault tolerant services. In: Proc. of the 19th ACM Symp. on Operating Systems Principles (SOSP). 2003. 253−267. [doi: 10.1145/945445.945470]

[55] Sousa J, Bessani A, Vukolić M. A Byzantine fault-tolerant ordering service for the Hyperledger Fabric blockchain platform. In: Proc. of the 48th Annual IEEE/IFIP Int'l Conf. on Dependable Systems and Networks (DSN). 2018. 51−58. [doi: 10.1109/DSN. 2018.00018]

[56] Sousa J, Bessani A. Separating the WHEAT from the chaff: An empirical design for geo-replicated state machines. In: Proc. of the 34th IEEE Symp. on Reliable Distributed Systems (SRDS). 2015. 146−155. [doi: 10.1109/SRDS.2015.40]

[57] Wood G. Ethereum: A Secure Decentralised Generalised Transaction Ledger. Ethereum Project Yellow Paper, 2014.

[58] Angelis SD, Aniello L, Baldoni R, Lombardi F, Margheri A, Sassone V. PBFT vs proof-of-authority: Applying the CAP theorem to permissioned blockchain. In: Proc. of the 2nd Italian Conf. on Cyber Security. 2018. 1−11.

[59] Merkle RC. A certified digital signature. In: Proc. of the 9th Annual Int'l Cryptology Conf. 1989. 218−238. [doi: 10.1007/0-387-34805-0_21]

[60] Prusty N. Building Blockchain Projects. Birmingham: Packt Publishing Ltd., 2017.

[61] Nguyen B, Cachin C, Yellick J, Androulaki E, Yang B, Caro AD, Christidis K, Vukolic M. Multichannel consensus. 2016. https://docs.google.com/document/d/1eRNxxQ0P8yp4Wh__Vi6ddaN_vhN2RQHP-IruHNUwyhc/edit#

[62] Carlyle J. CORDA performance: to infinity & beyond. 2018. https://www.r3.com/wp-content/uploads/2018/04/Corda-Performance-ENG.pdf

[63] Singh A, Das T, Maniatis P, Druschel P, Roscoe T. BFT protocols under fire. In: Proc. of the 5th USENIX Symp. on Networked Systems Design & Implementation (NSDI). 2008. 189−204.

[64] Croman K, Decker C, Eyal I, Gencer AE, Juels A, Ahmed E, Kosba AE, Miller A, Saxena P, Shi E, Sirer EG, Song D, Wattenhofer R. On scaling decentralized blockchains. In: Proc. of the Financial Cryptography and Data Security. 2016. 106−125. [doi: 10.1007/978-3-662-53357-4_8]

[65] Ben-Sasson E, Chiesa A, Genkin D, Tromer E, Virza M. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In: Proc. of the 33rd Annual Cryptology Conf., Vol.2. 2013. 90−108. [doi: 10.1007/978-3-642-40084-1_6]

[66] Ben-Sasson E, Chiesa A, Tromer E, Virza M. Succinct non-interactive zero knowledge for a von neumann architecture. In: Proc. of the 23rd USENIX Security Symp. 2014. 781−796.

[67] Zcash. ZSL integration: Proof of concept. Technical Report. Zerocoin Electric Coin Company, 2017.

[68] Microsoft. The Coco Framework Technical Overview. White Paper, 2017.

[69] Gentry C. Fully homomorphic encryption using ideal lattices. In: Proc. of the 41st Annual ACM Symp. on Theory of Computing (STOC). 2009. 169−178. [doi: 10.1145/1536414.1536440]

[70] Kosba A, Miller A, Shi E, Wen Z, Papamanthou C. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In: Proc. of the IEEE Symp. on Security and Privacy. 2016. 839−858. [doi: 10.1109/SP.2016.55]

[71] Wood G. Polkadot: Vision for a Heterogeneous Multi-Chain Farmework. White Paper, 2016.

[72] Kwon J, Buchman E. Cosmos: A Network of Distributed Ledgers. White Paper, 2016.

[73] Dinh TTA, Wang J, Chen G, Liu R, Ooi BC, Tan KL. Blockbench: A framework for analyzing private blockchains. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data (SIGMOD). 2017. 1085−1100. [doi: 10.1145/3035918.3064033]

[74] Atzei N, Bartoletti M, Cimoli T. A survey of attacks on ethereum smart contracts. In: Proc. of the 6th Int'l Conf. on Principles of Security and Trust (POST). 2017. 164−186. [doi: 10.1007/978-3-662-54455-6_8]

[75] Bhargavan K, Delignat-Lavaud A, Fournet C, Gollamudi A, Gonthier G, Kobeissi N, Kulatova N, Rastogi A, Sibut-Pinote T, Swamy N, Béguelin SZ. Formal verification of smart contracts. In: Proc. of the 2016 ACM Workshop on Programming Languages and Analysis for Security (PLAS). 2016: 91−96. [doi: 10.1145/2993600.2993611]

附中文参考文献:

[21] 袁勇,王飞跃.区块链技术发展现状与展望.自动化学报,2016,42(4):481−494. [doi: 10.16383/j.aas.2016.c160158]

[22] 何蒲,于戈,张岩峰,鲍玉斌.区块链技术与应用前瞻综述.计算机科学,2017,44(4):1−7. [doi: 10.11896/j.issn.1002-137X.2017.04.001]

[25] 钱卫宁,邵奇峰,朱燕超,金澈清,周傲英.区块链与可信数据管理:问题与方法.软件学报,2018,29(1):150−159. http://www.jos.org.cn/1000-9825/5434.htm [doi: 10.13328/j.cnki.jos.005434]

[26] 邵奇峰,金澈清,张召,钱卫宁,周傲英.区块链技:架构及进展.计算机学报,2018,41(5):969−988. [doi: 10.11897/SP.J.1016.2018.00969]

[41] 杨保华,陈昌.区块链原理、设计与应用.北京:机械工业出版社,2017.

[49] 张增骏,董宁,朱轩彤,陈剑雄.深度探索区块链:Hyperledger 技术与应用.北京:机械工业出版社,2018.

邵奇峰(1976－),男,河南郑州人,副教授,主要研究领域为区块链系统,大数据管理系统.

张召(1977－),女,博士,副教授,主要研究领域为区块链系统研发,海量数据管理与分析.

朱燕超(1992－),男,博士生,主要研究领域为区块链,分布式数据库.

周傲英(1965－),男,博士,教授,博士生导师,CCF 会士,主要研究领域为 Web 数据管理,数据密集型计算,内存集群计算,分布事务处理,大数据基准测试和性能优化.