区块链技术:架构及进展

邵奇峰[1),2)] 金澈清[1)] 张 召[1)] 钱卫宁[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 郑州 450007)

摘 要 传统的数据库管理系统主要由单一机构管理和维护,在多方参与者协作的场景中,因无法完全信任数据库中的数据,每方都需要单独构建一套承载自己业务数据的数据库,多方数据库间的数据差异会导致繁琐的人工对账和争议,而区块链可解决这种多方间的信任问题.区块链作为一种去中心化、不可篡改、可追溯、多方共同维护的分布式数据库,可在互不了解的多方间建立可靠的信任,在没有第三方中介机构的协调下,划时代地实现了可信的数据共享和点对点的价值传输.该文结合比特币、以太坊和 Hyperledger Fabric 等区块链平台,提出了区块链系统的体系架构;从区块链数据、共识机制、智能合约、可扩展性、安全性几个方面阐述了区块链的原理与技术;通过与传统数据库对比,总结了区块链的优势、劣势及发展趋势.

关键词 区块链;共识机制;智能合约;比特币;以太坊;hyperledger fabric

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2018.00969

Blockchain: Architecture and Research Progress

SHAO Qi-Feng[1),2)] JIN Che-Qing[1)] ZHANG Zhao[1)] QIAN Wei-Ning[1)] ZHOU Ao-Ying[1)]

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

[2)](*School of Software, Zhongyuan University of Technology, Zhengzhou* 450007)

Abstract Traditional database management systems are controlled by a single entity, because the data in the databases cannot be fully trusted by all participants in a multi-party collaboration scenario, each participant must maintain a separate database that hosts its own business data, the discrepancies on each participant's databases lead to slowdowns of manual reconciliation and some disputes. Blockchain can solve the problem of trust among multiple participants. Blockchain is a decentralized, trustless, tamper-proof and traceable distributed database managed by multiple participants. A blockchain also called distributed ledger, is essentially an append-only data structure maintained by a set of nodes which do not fully trust each other. Nodes in the blockchain keep replicas of the blocks, each containing an ordered set of transactions modifying the states. All nodes agree on the transactions and their order. Traditional databases assume a trusted environment. Blockchain's key property is that it assumes nodes behave in arbitrary manner. Being able to tolerate Byzantine failure by consensus protocol, blockchain establishes a reliable trust between both parties of the transaction, and implements trusted data sharing and peer-to-peer value exchange without third-party intermediaries. The blockchain's consensus protocol must tolerate Byzantine failures. This is not the case in traditional distributed systems,

收稿日期:2017-06-30;在线出版日期:2017-11-15.本课题得到国家"九七三"重点基础研究发展规划项目基金(2012CB316203)、国家自然科学基金(61432006,61370101)、河南省科技攻关计划项目(172102310714,172102210593)、河南省高等学校重点科研项目(15A520112)资助.**邵奇峰**,男,1976 年生,硕士,副教授,主要研究方向为大数据、区块链.E-mail: shao@zzti.edu.cn.**金澈清**,男,1977 年生,博士,教授,博士生导师,主要研究领域为基于位置的服务、不确定数据管理、数据流管理、数据质量分析和数据管理系统评测基准.**张 召**,女,1977 年生,博士,副教授,研究方向为海量数据管理、数据挖掘.**钱卫宁**,男,1976 年生,教授,博士生导师,主要研究领域为 Web 数据管理、社交媒体分析.**周傲英**,男,1965 年生,博士,教授,博士生导师,主要从事数据管理及应用研究,研究领域包括 Web 数据管理、数据密集型计算、内存集群计算、大数据基准测试和性能优化.

in which they use the Paxos or Raft consensus protocol. There are many variants of consensus protocols being developed for blockchains. They can be classified into two kinds. One consists of purely computation based protocols that use proof of computation to randomly select a node which decides the next block. PoW is the prime example. The other are purely communication based protocols in which nodes have equal votes and go through multiple rounds of communication to reach consensus. PBFT is the prime example. A blockchain system can be categorized as either public or consortium. In the former, any node can join and leave the system. In the latter, the blockchain enforces strict membership. There is an access control mechanism to determine who can join the system. PoW are used in public blockchains because they are fully decentralized, and PBFT are used in consortium blockchains because they assume authenticated nodes. Bitcoin is the most successful blockchain application. Ethereum is the most widely used platform in public blockchains. Hyperledger Fabric is the most widely used platform in consortium blockchains. This paper proposes an architecture model of the blockchain system based on the above three mainstream blockchain platforms. This paper then discusses the principles and technologies of blockchain according to blockchain data, consensus mechanism, smart contract, scalability and security. Specifically, blockchain data is introduced from three aspects of data structure, data model and datastore, consensus mechanism is discussed about PoW applied in public blockchains and PBFT applied in consortium blockchains, smart contract is described from three aspects of running mechanism, programing language and sandbox, blockchain scalability is analyzed about sharding and multichannel, blockchain security is discussed from digital signing and verification, and privacy preserving. This paper also analyzes the advantages, disadvantages and technology trends of blockchain by comparing with traditional databases, and gives several challenging research problems for blockchain. The development of blockchain will bring both challenges and opportunities for many industries, and result in the second generation of the digital revolution bringing us the Internet of Value.

Keywords blockchain; consensus mechanism; smart contract; Bitcoin; Ethereum; hyperledger fabric

1 引 言

传统的关系数据库管理系统、NoSQL 数据库管理系统都是由单一机构进行管理和维护，单一机构对所有数据拥有绝对的控制权，其它机构无法完整了解数据更新过程，因而无法完全信任数据库中的数据. 所以，在多个机构协作模式下，中心化的数据库管理系统始终存在信任问题. 以金融行业的清算和结算业务为例，传统中心化的数据库因无法解决多方互信问题，使得每个参与方都需要独立维护一套承载自己业务数据的数据库，这些数据库实际上是一座座信息孤岛，造成清结算过程耗费大量人工进行对账的情况，目前的清结算时间最快也需按天来计. 如果存在一个多方参与者一致信任的数据库系统，则可显著减少人工成本及缩短结算周期. 区块链(blockchain)是一种去中心化、不可篡改、可追溯、多方共同维护的分布式数据库，能够将传统单方维护的仅涉及自己业务的多个孤立数据库整合在一起，分布式地存储在多方共同维护的多个节点上，任何一方都无法完全控制这些数据，只能按照严格的规则和共识进行更新，从而实现了可信的多方间的信息共享和监督，避免了繁琐的人工对账，提高了业务处理效率，降低了交易成本. 区块链通过集成 P2P 协议、非对称加密、共识机制、块链结构等多种技术，解决了数据的可信问题. 通过应用区块链技术，无需借助任何第三方可信机构，互不了解、互不信任的多方可实现可信、对等的价值传输.

区块链源自于比特币(Bitcoin)的底层技术，2008 年，化名为“中本聪”(Satoshi Nakamoto)的学者提出了一种被称为比特币的数字货币，在没有任何权威中介机构统筹的情况下，互不信任的人可以

直接用比特币进行支付[1]. 2013 年 12 月，Buterin 提出了以太坊(Ethereum)区块链平台，除了可基于内置的以太币(Ether)实现数字货币交易外，还提供了图灵完备的编程语言以编写智能合约(smart contract)，从而首次将智能合约应用到了区块链[2]. 以太坊的愿景是创建一个永不停止、无审查、自动维护的去中心化的世界计算机. 2015 年 12 月，Linux 基金会发起了 Hyperledger① 开源区块链项目，旨在发展跨行业的商业区块链平台. Hyperledger 提供了 Fabric、Sawtooth、Iroha 和 Burrow 等多个区块链项目，其中最受关注的项目是 Fabric. 不同于比特币和以太坊，Hyperledge Fabric 专门针对于企业级的区块链应用而设计，并引入了成员管理服务[3]. 2016 年 4 月，R3 公司发布了面向金融机构定制设计的分布式账本平台 Corda[4]，该公司发起的 R3 联盟包括花旗银行、汇丰银行、德意志银行、法国兴业银行等 80 多家金融机构和监管成员，R3 声称 Corda 是受区块链启发的去中心化数据库，而不是一个传统的区块链平台，原因就是 R3 反对区块链中每个节点拥有全部数据，而注重保障数据仅对交易双方及监管可见的交易隐私性[5]. 2016 年 2 月，BigchainDB 公司发布了可扩展的区块链数据库 BigchainDB[6]，BigchainDB 既拥有高吞吐量、低延迟、大容量、丰富的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. BigchainDB 之所以具有分布式数据库的优点，是因为其底层数据库选用了 RethinkDB 数据库. 2017 年 1 月，国内的众享比特团队发布了号称全球首个基于区块链技术的数据库应用平台 ChainSQL[7]，ChainSQL 基于插件式管理，其底层支持 SQLite3、MySQL、PostgreSQL 等关系数据库. 2017 年 4 月腾讯发布了可信区块链平台 TrustSQL，致力于提供企业级区块链基础设施及区块链云服务. TrustSQL 支持自适应的共识机制、4000 TPS 的交易吞吐量、秒级交易确认及 Select、Insert 两种 SQL 语句[8]. 区块链平台可分为公有链和联盟链两类. 公有链中所有的节点可自由地加入或退出；而联盟链中的节点必须经过授权才可加入. 因此，公有链的节点通常是匿名的，而联盟链需要提供成员管理服务以对节点身份进行审核. 表 1 分别从准入机制、数据模型、共识算法、智能合约语言、底层数据库、数字货币几个方面对常用区块链平台进行了对比.

表 1　区块链平台对比

区块链平台	准入机制	数据模型	共识算法	智能合约语言	底层数据库	数字货币
Bitcoin	公有链	基于交易	PoW	基于栈的脚本	LevelDB	比特币
Ethereum	公有链	基于账户	PoW/PoS	Solidity/Serpent	LevelDB	以太币
Hyperledger Fabric	联盟链	基于账户	PBFT/SBFT	Go/Java	LevelDB/CouchDB	—
Hyperledger Sawtooth	公有链/联盟链	基于账户	PoET	Python	—	—
Corda	联盟链	基于交易	Raft	Java/Kotlin	常用关系数据库	—
Ripple	联盟链	基于账户	RPCA	—	RocksDB/SQLite	瑞波币
BigchainDB	联盟链	基于交易	Quorum Voting	Crypto-Conditions	RethinkDB/MongoDB	—
TrustSQL	联盟链	基于账户	BFT-Raft/PBFT	JavaScript	MySQL/MariaDB	—

数据库最初发展的原点是文件系统，为了满足以银行为代表的金融机构的业务需求，引发了关系数据库领域中的关系模型、事务处理、查询优化三大成就，产生了一系列的关系数据库产品. 后来，随着互联网行业的快速发展，非结构化数据的数据量已经远超结构化数据的数据量，从而引发了 NoSQL 数据库系统的发展，产生了一系列的 NoSQL 数据库产品. 如今，随着去中介的共享经济的发展，区块链作为一种去中心化的分布式数据库，解决了可信的价值传输问题，因此将成为共享经济业务的理想数据库平台.

比特币是最成功的区块链应用，但其仅限于数字货币类型的应用，为此业界推出了多种支持通用应用的区块链平台. 公有链中应用最广泛的通用平台是以太坊，Quorum②、Monax③、DFINITY④、HydraChain⑤ 和 BCOS⑥ 等众多区块链平台都是基于 Ethereum 构建和扩展. 联盟链中应用最广泛的通用平台是 Hyperledger Fabric，其拥有 IBM、Intel、J. P. Morgan、R3、DTCC、SWIFT 等 130 多名成员. 袁勇等人[9]阐述了区块链及比特币的原理、技术、方法与应用，展望了基于区块链的平行社会. 何蒲和于戈等人[10]基于比特币介绍了区块链基础技

① Hyperledger. https://www.hyperledger.org/projects
② Quorum. https://www.jpmorgan.com/quorum
③ Monax. https://monax.io
④ DFINITY. https://dfinity.network/tech.html
⑤ HydraChain. https://github.com/HydraChain/hydrachain
⑥ BCOS. https://github.com/bcosorg

术、概念及与应用前景. Tschorsch 等人[11]从协议、安全、网络和隐私等方面分析了比特币原理. 已有文献更偏重基于比特币来介绍区块链的研究现状，本文则结合比特币、以太坊和 Hyperledger Fabric 平台的共性与差异进行分析，在此基础上介绍区块链技术的研究进展. 本文第 2 节提出区块链的系统架构；第 3 节从数据结构、数据模型和数据存储三方面分析区块链数据；第 4 节阐述应用于公有链的 PoW 机制和应用于联盟链的 PBFT 算法；第 5 节从运作机制、编程语言、沙箱环境三方面论述了智能合约；第 6 节基于分片和多通道介绍区块链的可扩展性方案；第 7 节基于签名与验证、隐私保护介绍区块链目前的安全性方案；第 8 节总结区块链的优势、劣势及发展趋势；结束语提出区块链对传统行业的影响.

2 区块链体系架构

从最早应用区块链技术的比特币到最先在区块链引入智能合约的以太坊，再到应用最广的联盟链 Hyperledger Fabric，它们尽管在具体实现上各有不同，但在整体体系架构上存在着诸多共性. 如图 1 所示，区块链平台整体上可划分为网络层、共识层、数据层、智能合约层和应用层五个层次.

		比特币	以太坊	Hyperledger Fabric
应用层		比特币交易	Dapp/以太币交易	企业级区块链应用
智能合约层	编程语言	Script	Solidity/Serpent	Go/Java
	沙盒环境		EVM	Docker
数据层	数据结构	Merkle树/区块链表	Merkle Patricia树/区块链表	Merkle Bucket树/区块链表
	数据模型	基于交易的模型	基于账户的模型	基于账户的模型
	区块存储	文件存储	LevelDB	文件存储
共识层		PoW	PoW/PoS	PBFT/SBFT
网络层		TCP-based P2P	TCP-based P2P	HTTP/2-based P2P

图 1 区块链体系架构

2.1 网络层

2001 年，Gribble 等人[12]提出将 P2P 技术与数据库系统进行联合研究的想法. 早期的 P2P 数据库没有预定的全局模式，不能适应网络变化而查询到完整的结果集[13-14]，因而不适合企业级应用. 基于 P2P 的区块链则可实现数字资产交易类的金融应用，区块链网络中没有中心节点，任意两个节点间可直接进行交易，任何时刻每个节点也可自由加入或退出网络，因此，区块链平台通常选择完全分布式且可容忍单点故障的 P2P 协议作为网络传输协议. 区块链网络节点具有平等、自治、分布等特性，所有节点以扁平拓扑结构相互连通，不存在任何中心化的权威节点和层级结构，每个节点均拥有路由发现、广播交易、广播区块、发现新节点等功能[15].

区块链网络的 P2P 协议主要用于节点间传输交易数据和区块数据，比特币和以太坊的 P2P 协议基于 TCP 协议实现，Hyperledger Fabric 的 P2P 协议则基于 HTTP/2 协议实现. 在区块链网络中，节点时刻监听网络中广播的数据，当接收到邻居节点发来的新交易和新区块时，其首先会验证这些交易和区块是否有效，包括交易中的数字签名、区块中的工作量证明等，只有验证通过的交易和区块才会被处理(新交易被加入正在构建的区块，新区块被链接到区块链)和转发，以防止无效数据的继续传播.

2.2 共识层

分布式数据库主要使用 Paxos[16-17] 和 Raft[18] 算法解决分布式一致性问题，这些数据库都由单一机构管理维护，所有节点都是可信的，算法只需支持崩溃容错(Crash Fault-Tolerant，CFT). 去中心化的区块链由多方共同管理维护，其网络节点可由任何一方提供，部分节点可能并不可信，因而需要支持更为复杂的拜占庭容错(Byzantine Fault-Tolerant，BFT). 假设在总共 n 个节点的网络中至多包含 f 个不可信节点，对于同步通讯且可靠的网络而言，拜占庭将军问题能够在 $n \geq 3f+1$ 的条件下被解决[19]. 而如果是异步通讯，Fischer、Lynch 和 Paterson[20] 证明确定性的共识机制无法容忍任何节点失效. Castro 和 Liskov[21] 提出了 Practical Byzantine Fault Tolerance(PBFT)，将拜占庭协议的复杂度从指数级降低到多项式级别，使拜占庭协议在分布式系统中应用成为可能. 为了提升 PBFT 的性能，Kotla 等人[22] 提出了 Zyzzyva，认为网络节点在绝大部分时间都处于正常状态，无需在每个请求都达成一致后再执行，而只需在发生错误之后再达成一致. Kwon[23] 提出了 Tendermint，在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 2/3 即认为可达成共识. 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. Liu 等人[24] 提出了 Cross Fault Tolerance(XFT)，其认为恶意者很难同时控制整个网络和拜占庭节点，从而简化了 BFT 消息模式，

可在 $n \geq 2f+1$ 条件下解决拜占庭将军问题；此外，业界还提出了 Scalable BFT[25]、Parallel BFT[26]，Optimistic BFT[27] 等 BFT 改进算法. Ripple 支付网络提出了基于一组可信认证节点的 Ripple Protocol Consensus Algorithm(RPCA)，能够在 $n \geq 5f+1$ 条件下解决拜占庭将军问题[28].

为了解决节点自由进出可能带来的女巫攻击(sybil attack)[29] 问题，比特币应用了工作量证明(Proof of Work，PoW)机制. PoW 源自于 Dwork 等人[30] 防范垃圾邮件的研究工作，即只有完成了一定计算工作并提供了证明的邮件才会被接收. Back 提出了 Hashcash①，其是一种基于哈希函数的工作量证明算法. 比特币要求只有完成一定计算工作量并提供证明的节点才可生成区块，每个网络节点利用自身计算资源进行哈希运算以竞争区块记账权，只要全网可信节点所控制的计算资源高于 51%，即可证明整个网络是安全的[31]. 为了避免高度依赖节点算力所带来的电能消耗，研究者提出一些不依赖算力而能够达成共识的机制. 点点币(Peercoin)应用了区块生成难度与节点所占股权成反比的权益证明(Proof of Stake，PoS)机制[32]；比特股(Bitshares)应用了获股东投票数最多的几位代表按既定时间段轮流产生区块的股份授权证明(Delegated Proof of Stake，DPoS)机制[33]. Hyperledger Sawtooth 应用了基于 Intel SGX② 可信硬件的逝去时间证明(Proof of Elapsed Time，PoET③)机制.

基于证明机制的共识通常适用于节点自由进出的公有链，比特币与以太坊使用 PoW 机制；基于投票机制的共识则通常适用于节点授权加入的联盟链，Hyperledger Fabric 使用 PBFT 算法.

2.3　数据层

比特币、以太坊和 Hyperledger Fabric 在区块链数据结构、数据模型和数据存储方面各有特色.

在数据结构的设计上，现有区块链平台借鉴了 Haber 与 Stornetta[34-36] 的研究工作，他们设计了基于文档时间戳的数字公证服务以证明各类电子文档的创建时间. 时间戳服务器对新建文档、当前时间及指向之前文档签名的哈希指针进行签名，后续文档又对当前文档签名进行签名，如此形成了一个基于时间戳的证书链，该链反映了文件创建的先后顺序，且链中的时间戳无法篡改. Haber 与 Stornetta 还提出将多个文档组成块并针对块进行签名、用 Merkle 树[37-39] 组织块内文档等方案. 区块链中每个区块包含区块头和区块体两部分，区块体存放批量交易数据，区块头存放 Merkle 根、前块哈希、时间戳等数据. 基于块内交易数据哈希生成的 Merkle 根实现了块内交易数据的不可篡改性与简单支付验证；基于前一区块内容生成的前块哈希将孤立的区块链接在一起，形成了区块链；时间戳表明了该区块的生成时间. 比特币的区块头还包含难度目标、Nonce 等数据，以支持 PoW 共识机制中的挖矿运算.

在数据模型的设计上，比特币采用了基于交易的数据模型，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；以太坊与 Hyperledger Fabric 需要支持功能丰富的通用应用，因此采用了基于账户的模型，可基于账户快速查询到当前余额或状态[40].

在数据存储的设计上，因为区块链数据类似于传统数据库的预写式日志，因此通常都按日志文件格式存储；由于系统需要大量基于哈希的键值检索(如基于交易哈希检索交易数据、基于区块哈希检索区块数据)，索引数据和状态数据通常存储在 Key-Value 数据库，如比特币、以太坊与 Hyperledger Fabric 都以 LevelDB④ 数据库存储索引数据.

2.4　智能合约层

智能合约是一种用算法和程序来编制合同条款、部署在区块链上且可按照规则自动执行的数字化协议. 该概念早在 1994 年由 Szabo 提出[41]，起初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. 由于早期计算条件的限制和应用场景的缺失，智能合约并未受到研究者的广泛关注，直到区块链技术出现之后，智能合约才被重新定义. 区块链实现了去中心化的存储，智能合约在其基础上则实现了去中心化的计算.

比特币脚本是嵌在比特币交易上的一组指令，由于指令类型单一、实现功能有限，其只能算作智能合约的雏形. 以太坊提供了图灵完备的脚本语言 Solidity⑤、Serpent⑥ 与沙盒环境 Ethereum Virtual Machine(EVM)[42]，以供用户编写和运行智能合约. Hyperledger Fabric 的智能合约被称为 Chaincode，其选用 Docker 容器作为沙盒环境，Docker 容器中

① Hashcash. http://www.cypherspace.org/hashcash/hashcash.pdf
② Intel SGX. https://software.intel.com/en-us/sgx
③ PoET. https://sawtooth.hyperledger.org/docs/core/releases/latest/introduction.html#proof-of-elapsed-time-poet
④ LevelDB. https://github.com/google/leveldb
⑤ Solidity. http://solidity.readthedocs.io
⑥ Serpent. https://github.com/ethereum/wiki/wiki/Serpent

带有一组经过签名的基础磁盘映像及 Go 与 Java 语言的运行时和 SDK，以运行 Go 与 Java 语言编写的 Chaincode[43].

2.5 应用层

比特币平台上的应用主要是基于比特币的数字货币交易. 以太坊除了基于以太币的数字货币交易外，还支持去中心化应用(Decentralized Application，Dapp)，Dapp 是由 JavaScript 构建的 Web 前端应用，通过 JSON-RPC 与运行在以太坊节点上的智能合约进行通信. Hyperledger Fabric 主要面向企业级的区块链应用，并没有提供数字货币，其应用可基于 Go、Java、Python、Node. js 等语言的 SDK 构建[44]，并通过 gRPC 或 REST 与运行在 Hyperledger Fabric 节点上的智能合约进行通信.

3 区块链数据

3.1 区块链数据结构

为了实现数据的不可篡改性，区块链引入了以区块为单位的链式结构. 不同区块链平台在数据结构的具体细节虽有差异，但整体上基本相同. 以比特币为例，每个区块由区块头和区块体两部分组成，区块体中存放了自前一区块之后发生的多笔交易；区块头中存放了前块哈希(PrevBlockHash)、随机数(Nonce)、Merkle 根(Merkle Root)等，详细结构如表 2 所示.

表 2 比特币区块头部结构

字段	描述	大小
版本	比特币软件/协议的版本	4 字节
前块哈希	前一区块的哈希值	32 字节
Merkle 根	该区块中所有交易计算出的 Merkle 根	32 字节
时间戳	该区块产生的近似时间	4 字节
难度目标	生成该区块的难度目标	4 字节
随机数	使区块头部哈希值小于难度目标的整数	4 字节

区块链基于两种哈希结构保障了数据的不可篡改性，即 Merkle 树和区块链表，图 2 描述了比特币的区块链数据结构.

(1) Merkle 树. Ralph Merkle 提出的 Merkle 树原用于生成数字证书目录的摘要，后来提出了很多种改进，比特币使用了最简单的二叉 Merkle 树. 树上的每个结点都是哈希值，每个叶子结点对应块内一笔交易数据的 SHA256 哈希；两个子结点的值连接之后，再经哈希运算可得到父结点的值；如此反复执行两两哈希，直至生成根哈希值，即交易 Merkle 根. 通过 Merkle 根，块内任何交易数据的篡改都会被检测到，从而确保交易数据的完整性. 无需树上其它结点参与，仅根据交易结点到 Merkle 根路径上的直接分支，即可基于简单支付验证(Simplified Payment Verification，SPV)确认一个交易是否存在于该块. 例如，仅需图 2 中的结点 Hash3、Hash12 和 Merkle 根即可验证交易 Tx4 是否位于该块. 在由 N 个交易组成的区块中，至多计算 $2\log_2(N)$ 次哈希即可验证交易是否存在. 不需全部区块数据即可验证交易，使其非常适合于构建轻客户端和电子钱包.

以太坊和 Hyperledger Fabric 块头除含有交易 Merkle 根外，还含有针对账户状态数据的状态 Merkle 根(State Root)，以太坊块头还含有针对交易执行日志的收据 Merkle 根(Receipts Root). 以太坊计算 Merkle 根使用的是 Merkle Patricia 树[45]，虽然区块中的交易数据是不变的，但状态数据经常改变且数量众多，构建新区块时，Merkle Patricia 树仅需计算在新区块中变化了的账户状态，状态没有变化的分支可直接引用，而无需重新计算整棵树. 如图 3 所示，Merkle Patricia 树包含扩展结点、分支结点和叶子结点. 扩展结点包含了共同的 Key 前缀；分支结点通常在扩展结点之后，基于单个 16 进制字符的 Key 前缀实现了树的分支；叶子结点包含一个

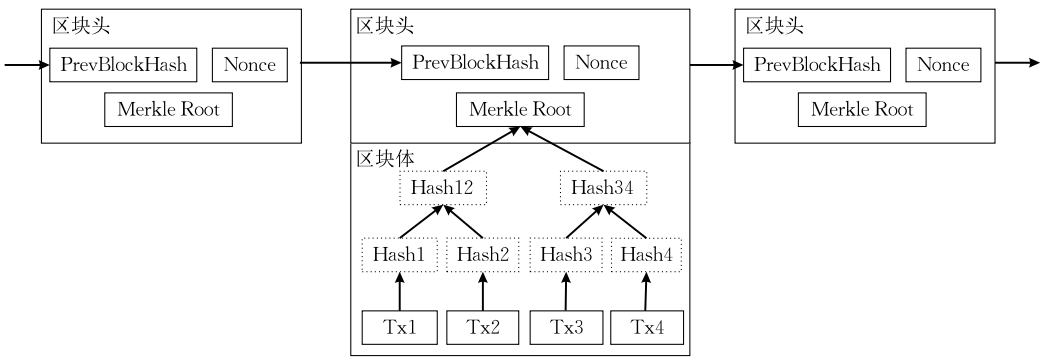

图 2 比特币区块链的两种哈希结构

以太坊账户状态. Merkle Patricia 树实质上是融合了 Merkle 树和前缀树，因此其具有查找能力. 以一个以太坊账户地址为查找路径，能够快速的从 Merkle Patricia 树根向下查找到叶子结点中账户的状态数据，这种查找能力是二叉 Merkle 树所不具备的. Merkle Patricia 树还具有深度有限、根值与结点更新顺序无关等特性. Hyperledger Fabric 计算状态 Merkle 根使用的是 Merkle Bucket 树，Merkle Bucket 树是多叉树，每个叶子结点是一个桶，桶中存放的是 Key-Value 类型的状态数据集. 为新区块计算状态根时，没有变化的桶可以被跳过，因而可快速计算状态根. Merkle Bucket 树可通过调整桶数和分支数来控制树的深度和宽度，从而可在不同的性能和资源需求间权衡.

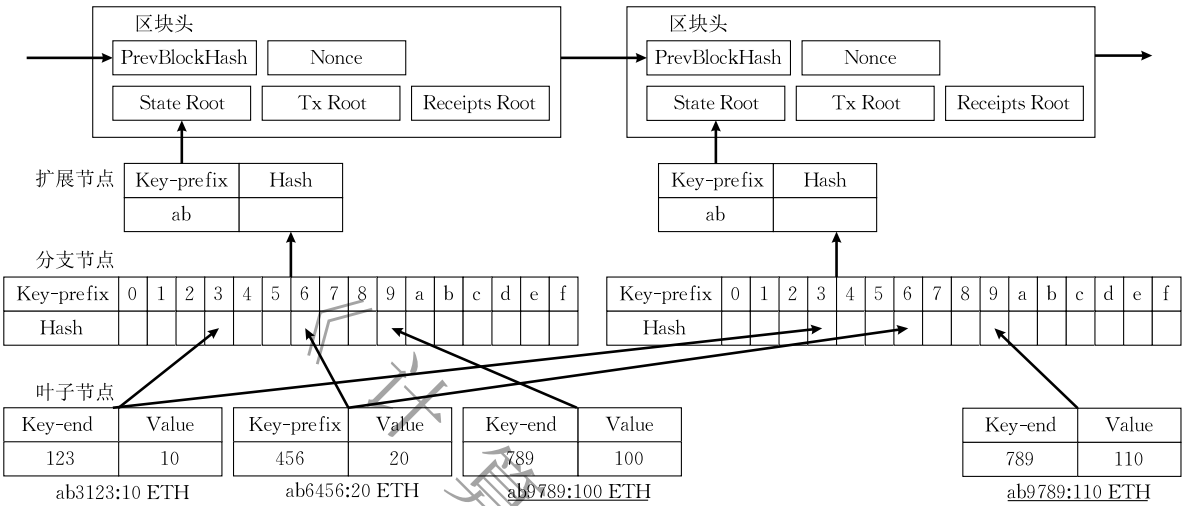

图 3　Merkle Patricia 树

(2) 区块链表. 对区块头中的前块哈希(PrevBlockHash)、随机数(Nonce)和 Merkle 根等元数据进行两次 SHA256 哈希运算即可得到该区块的块哈希. 如图 2 所示，PrevBlockHash 存放前一区块的块哈希，所有区块按照生成顺序以 PrevBlockHash 为哈希指针链接在一起，就形成了一条区块链表. 区块头包含交易 Merkle 根，所以通过块哈希可以验证区块头部和区块中的交易数据是否被篡改；区块头还包含前块哈希 PrevBlockHash，所以通过块哈希还可验证该区块之前直至创世区块的所有区块是否被篡改. 依靠前块哈希指针 PrevBlockHash，所有区块环环相扣，任一区块若被篡改，都会引发其后所有区块哈希指针的连锁改变. 当从不可信节点下载某块及之前所有块时，基于块哈希可验证各块是否被修改过.

3.2　区块链数据模型

比特币区块链采用基于交易的模型；以太坊、Hyperledger Fabric 区块链采用了基于账户的模型.

(1) 基于交易的模型. 以数字货币为基础的区块链中的交易通常就是转账，图 4 为比特币中交易的数据结构. 每个交易由交易输入和交易输出组成，交易输入和交易输出可以有多项，表示一次交易可以将先前多个账户中的比特币合并后转给另外多个

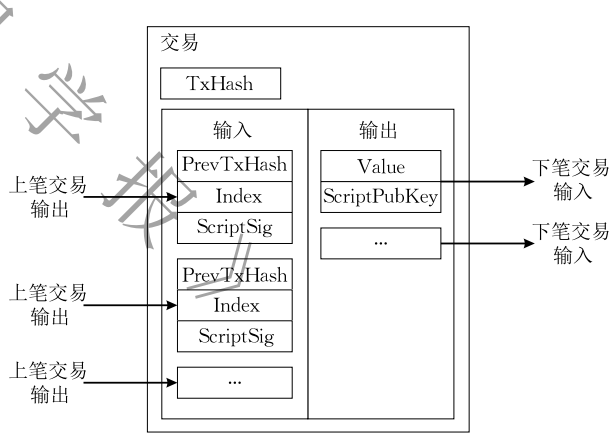

图 4　比特币交易的数据结构

账户. 交易输入的结构如表 3 所示，每个输入主要由上笔交易的哈希 PrevTxHash、上笔交易的输出索引 Index 和输入脚本 ScriptSig 组成. 每笔交易的输入都对应于某笔历史交易的输出，PrevTxHash 哈希指针是之前某笔历史交易的哈希值；Index 指出输入对应于该历史交易的第几个交易输出，脚本 ScriptSig 包含了比特币持有者对当前交易的签名. 交易输出的结构如表 4 所示，每个输出包括转账金额 Value 和包含接收者公钥哈希的脚本 ScriptPubKey. 在基于交易的模型中，所有交易依靠上笔交易哈希指针构成了多条以交易为结点的链表，每

笔交易可一直向前追溯至源头的 Coinbase(即挖矿所得的比特币),向后可追踪至尚未花费的交易.如果一笔交易的输出没有任何另一笔交易的输入与之对应,则说明该输出中的比特币未被花费.通过收集当前所有未花费的交易输出(Unspent Transaction Outputs,UTXO),可快速验证某交易中的比特币是否已花费.针对某一比特币地址,其所有未花费交易的比特币之和,即为该账户的比特币余额.基于交易的模型可有效地防范伪造、双花等针对数字货币的攻击.

表 3 比特币交易输入的结构

字段	描述	大小
上笔交易哈希	指向上笔交易,该交易包含了需被支付的比特币	32 字节
上笔交易输出索引	指出上笔交易的第几个交易输出,从 0 开始计数	4 字节
输入脚本尺寸	以字节为单位	1~9 字节
输入脚本	包含了花费比特币所需的证明,如:持有者的签名	可变
序列号	未启用,缺省为 0xFFFFFFFF	4 字节

表 4 比特币交易输出的结构

字段	描述	大小
转账金额	以聪(10^{-8}Bitcoin)为单位	8 字节
输出脚本尺寸	以字节为单位	1~9 字节
输出脚本	定义了比特币被花费的条件,如:接收者的比特币地址	可变

(2)基于账户的模型.基于交易的模型虽可方便地验证交易,但却无法快速查询用户余额.为了支持更多类型的行业应用,以太坊、Hyperledger Fabric 等区块链平台采用了基于账户的模型,从而可方便地查询交易余额或业务状态数据.智能合约也更适合于在基于账户的模型之上构建,其针对状态数据更易处理复杂的业务逻辑.以太坊下的账户分为外部账户(Externally Owned Account)和合约账户(Contract Account)两种类型.外部账户用于表达一个普通账户的以太币余额,合约账户用于表达一个以太坊智能合约,普通账户中的余额、智能合约中的状态变量都属于以太坊状态数据.图 5 描述了以太坊账户的状态转换过程,状态反映了账户中各属性的当前值,涉及账户的一笔交易发生时,会引起账户状态的变化.外部账户和合约账户在以太坊下用同一数据结构表示,其包含 Balance、Nonce、CodeHash 和 StorageRoot 四个属性,Balance 是账户中的以太币余额;Nonce 是对账户发送过的交易的计数,用于防范重放攻击;当账户被应用于智能合约时,CodeHash 为合约代码的哈希值,StorageRoot 是合约状态数据的 Merkle Patricia 树根.以太坊的交易包含 To、Value、Nonce、gasPrice、gasLimit、Data 及交易签名七个属性.To 是接收者的账户地址,Value 是转账的以太币金额,Nonce 是发送者对本次交易的计数,gasPrice 是交易时 Gas 的以太币单价,gasLimit 是执行该交易所允许消耗的最大 Gas 数额,Data 是调用智能合约时的消息数据,交易签名是发送者对交易的 ECDSA 签名.

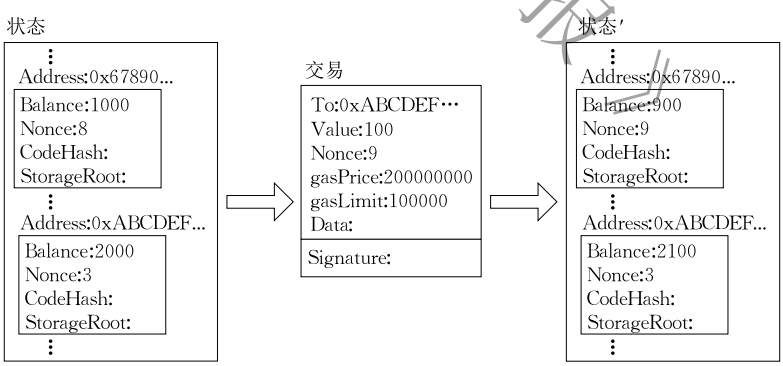

图 5 以太坊账户的状态转换

3.3 区块链数据存储

区块链在磁盘上既可以文件形式存储,也可以数据库形式存储.文件存储更方便于日志形式的追加操作,数据库存储更易实现查询与修改.比特币、Hyperledger Fabric 1.0 的区块链以文件形式存储,其索引数据存储在 LevelDB 数据库;以太坊的区块链与索引都存储在 LevelDB 数据库.区块链系统中存在大量哈希计算,交易、区块都依靠哈希值进行标识,所以底层数据库通常都选择了 Key-Value 数据库.

除了存储区块链数据,以太坊、Hyperledger Fabric 还都基于 LevelDB 构建了状态数据库(world state)以存储账户余额或业务状态数据.在基于账户模型的区块链平台中,交易数据被打包进区块且经共识算法确认后,先追加入区块链,而后再写入状态数据库.当新的节点加入区块链网络时,为了和已有节点数据保持一致,新节点需要同步区块链数据

以达到全网区块的高度，同时，状态数据库也需和全网同步. 从传统数据库来看，区块链数据相当于状态数据库的 WAL(Write-Ahead Logging)[46] 日志.

为了让更多常规计算能力的个人计算机节点加入到区块链网络，以充分实现去中心化的理念，比特币和以太坊都选择了 LevelDB 数据库存储索引或状态数据，因为 LevelDB 是轻量级的单机数据库，无需安装部署且写入性能高效. 但是基于 LevelDB 数据库的架构方案显然无法满足企业级的业务需求(如高并发访问、Non-Key 查询等)，因此，Hyperledger Fabric 1.0 提供了插件化的数据访问机制，其除了支持 LevelDB 数据库，还额外支持 CouchDB[47] 分布式数据库.

4　共识机制

传统分布式数据库主要使用 Paxos 和 Raft 算法解决分布式一致性问题，它们假定系统中每个节点都是忠诚、不作恶的，但报文可能发生丢失和延时等问题. 当分布式数据库的所有节点由单一机构统一维护时，此假定成立. 在去中心化的区块链网络中，节点由互不了解、互不信任的多方参与者共同提供和维护，受各种利益驱动，网络中的参与者存在欺骗、作恶的可能，因此 Paxos 和 Raft 算法不能直接用于区块链的共识. 本节分别介绍适用于公有链的 PoW 机制和适用于联盟链的 PBFT 算法，表 5 给出了 PoW 和 PBFT 在各种特性上的对比.

表 5　PoW 与 PBFT 对比

	PoW	PBFT
节点准入机制	公有链	联盟链
交易吞吐量	7 TPS(比特币) 20～30 TPS(以太坊)	200～2000 TPS (Fabric 0.6)
交易确认时间	60 min(比特币) 3 min(以太坊)	毫秒级(Fabric 0.6)
可扩展性	节点数＜100000	节点数＜100
拜占庭容错	＜50%的算力	＜33%的投票
资源消耗	哈希消耗计算资源	广播消耗网络资源
分叉可能性	有	无
最终一致性	否	是

4.1　PoW

公有链中每个节点身份匿名且可自由进出，使得基于节点投票机制的共识算法不适合公有链，因为恶意攻击者可创建任意多节点以增加投票权，从而发动女巫攻击并误导系统. PoW 机制可有效应对女巫攻击，其依靠分布式节点间的算力竞争来保证全网区块链数据的一致性和安全性. PoW 机制要求每个节点基于自身算力解决求解复杂但验证容易的 SHA256 计算难题，即寻找一个合适的随机数 Nonce，使得区块头部元数据的 SHA256 哈希值小于区块头中难度目标的设定值：

$$H(n \| h) \leq t \tag{1}$$

H 为 SHA256 哈希函数；n 为随机数 Nonce；h 为区块头部数据，主要包含前块哈希、Merkle 根等内容；t 为难度目标，t 值越小，n 值越难找到；最先寻找到的节点可获得新区块的记账权. PoW 在区块链网络中的共识流程如下：

(1) 每笔新交易被广播到区块链网络的所有节点.

(2) 为了构建新的区块，每个节点收集自前一区块生成以来接收到的所有交易，并根据这些交易计算出区块头部的 Merkle 根. 将区块头部的随机数 Nonce 从 0 开始递增加 1，直至区块头的两次 SHA256 哈希值小于或等于难度目标的设定值为止.

(3) 全网节点同时参与计算，若某节点先找到了正确的随机数，则该节点将获得新区块的记账权及奖励(奖励包括新区块中的区块奖励及每笔交易的交易费用)，并将该区块向全网广播.

(4) 其它节点接收到新区块后，验证区块中的交易和随机数 Nonce 的有效性，如果正确，就将该区块加入本地的区块链，并基于该块开始构建下一区块.

PoW 机制将经济激励与共识过程相融合，促使更多节点参与挖矿并保持诚信，从而主动增强了网络的可靠性与安全性，这是其它共识算法不具备的. 挖矿实质是寻找由多个前导零构成的区块头哈希值，当难度目标的设定值越小，区块头哈希值的前导零就越多，寻找到合适随机数的概率越低，挖矿的难度就越大. 为了适应硬件技术的快速发展及计算能力的不断提升，比特币每 2016 块就会调整一次难度目标，以控制区块的平均生成时间(10 min)始终保持不变. 对于 PoW 机制而言，若要篡改和伪造链中某一区块，就必须针对该区块及其后每个区块重新寻找块头的随机数 Nonce，并且计算速度还要超过主链，这需要至少掌握全网 51%以上的算力，才能使攻击成为可能，因此攻击的难度和成本非常高. PoW 机制实质上是以牺牲性能换取了数据的一致性和安全性，所以基于 PoW 机制的区块链平台的性能相对较低. 目前比特币平均每 10 min 产生一个区块，区块尺寸上限为 1 MB，普通交易(包含一个输入与两个输出)的尺寸约为 250 B，可计算出交易

吞吐量约为 7 TPS[48]；以太坊平均每 12～15 s 产生一个区块，交易吞吐量约为 20～30 TPS.

决定交易吞吐量的两个关键参数是区块尺寸和出块间隔. 大区块能容纳更多交易，但在网络中传播需要更长时间，会增加分叉的风险；大区块还需要更强算力的矿机，会增加中心化的风险. 减小出块间隔，就需降低挖矿难度，其会使网络中同时挖矿成功的节点增多，若多个区块同时被创建就会引起分叉. 当发生分叉时，最长的链即花费了最多算力的链被认为是主链，其它则被认为是分支，分支中的所有交易会被忽略. 分叉不但增加了有效块的作废率，还易引起双花攻击. 比特币将分支结点上的区块称为孤块，并会将其作为废块而丢弃. 为了保证挖矿的公平性及避免挖矿算力的浪费，以太坊引入了 GHOST (Greedy Heaviest-Observed Sub-Tree)[49]协议来处理分叉，GHOST 协议认为分支上的有效区块对确认主链上的交易也有贡献，因而没有丢弃该区块，而是将该区块作为叔块并给予相当主块 87.5%的奖励，给予叔块的直接子块相当主块 12.5%的奖励，矿工每引用一个叔块给予相当主块 3%的奖励. 分叉会带来交易的不确定性，目前比特币假定 6 个区块之后，交易已基本不可逆，所以其交易确认时间为 60 min；以太坊假定 12 个区块之后，交易已基本不可逆，所以其交易确认时间为 3 min；但理论上这些交易并未最终确认.

针对 PoW 机制低效、耗能的缺陷，PoS 机制无需矿工购买矿机、消耗电力及进行无意义的计算，而是根据矿工在区块链中拥有的股权(即数字货币量)来决定其挖矿的难度：

$$H(n \| h) \leq s(M).t \tag{2}$$

M 为某矿工；函数 s 返回该矿工拥有的股权；当矿工 M 拥有的股权越多，挖矿的整体难度就会越低，其越容易找到合适的 n. 以太坊基于 PoS 机制提出了 Casper① 共识，需要矿工购买以太币并注入以太坊作为抵押. Casper 以智能合约的方式实现，根据抵押的以太币数量和时间，成比例的分配区块的记账权和奖励. 与 PoS 机制不同，Casper 还引入了惩罚措施，一旦发现某个矿工作弊，其抵押的所有以太币将全被罚没，参与共识和出块的权利也会被取消. Casper 虚拟了比特币的挖矿过程，可使出块时间减少到 4 s，且无需进行挖矿及消耗额外的电力.

4.2 PBFT

PoW 机制依靠算力竞争来保障区块链数据的一致性及安全性，但算力竞争毫无意义地消耗了大量计算资源和电力能源，并不适合于针对企业级应用的联盟链. 联盟链更适合应用无需消耗计算资源和电力能源的 PBFT 算法. PBFT 算法可容忍恶意节点不超过全网节点数量的 1/3，即如果有超过 2/3 的正常节点，就可保障数据的一致性和安全性. PBFT 在区块链网络中的共识流程如下：

(1) 从全网节点选举出一个主节点，新区块由主节点负责生成.

(2) 每个节点把新交易向全网广播，主节点把从网络收集到的需放在新区块内的多个交易排序后存入列表，并将该列表向全网广播.

(3) 每个节点接收到交易列表后，依据排序模拟执行交易. 所有交易执行完后，基于交易结果计算新区块的哈希摘要，并向全网广播.

(4) 如果一个节点收到的 $2f$(f 为可容忍的恶意节点数)条其它节点发来的摘要都和自己相同，就向全网广播一条 commit 消息.

(5) 如果一个节点收到 $2f+1$ 条 commit 消息，即可正式提交新区块及其交易到本地的区块链和状态数据库.

PBFT 共识过程中的每个区块都由唯一的主节点生成，所以不存在分叉的可能. 但在节点数为 N 的网络中，该算法有两个阶段需要传输的网络消息为 $O(N^2)$，其会造成很大的网络开销. 因此，目前基于 PBFT 算法的区块链的系统性能并不高[50]. 另外，如何支持共识节点的动态加入和退出也是有待解决的问题. 目前，Hyperledger Fabric 1.0 正在基于插件化开发 BFT-SMART[51]、Simplified Byzantine Fault Tolerance (SBFT)② 和 HoneyBadgerBFT[52] 等共识模块. BFT-SMART 是一个强调实用性、可靠性与模块化的 BFT 软件库，其具有多核感知、节点动态进出可配置等特性. SBFT 假设主节不会成为恶意节点，从而构建了类似于 Raft 的消息模式，减少了网络中的广播通信. HoneyBadgerBFT 是首个实用的异步 BFT 协议，其不依赖任何时间假设就可保证网络有效性，即使在网络行为无法预测的广域网中也可容错.

5 智能合约

智能合约是用程序语言编写的商业合约，在预

① Introducing Casper: The friendly ghost. https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost/

② Chain: Simplified Byzantine Fault Tolerance (SBFT). http://sammantics.com/blog/2016/7/27/chain-1

定条件满足时，能够自动强制的执行合同条款，实现“代码即法律”的目标. 区块链的去中心化使得智能合约在没有中心管理者参与的情况下，可同时运行在全网所有节点，任何机构和个人都无法将其强行停止. 比特币平台并不支持智能合约，本节主要围绕以太坊和 Hyperledger Fabric 介绍智能合约.

5.1　运作机制

智能合约是运行在区块链上的一段计算机程序，其扩展了区块链的功能，丰富了区块链的上层应用，智能合约的运作机制如图 6 所示. 依照商业逻辑编写完智能合约代码后，需要将其发布到区块链网络节点上. 在以太坊中，部署后的合约存放在区块链上，每次被调用时才被以太坊虚拟机(EVM)加载运行；在 Hyperledger Fabric 中，部署后的合约被打包成 Docker 镜像，每个节点基于该镜像启动一个新的 Docker 容器并执行合约中的初始化方法，然后等待被调用. 外部应用通过调用智能合约来实现各种交易，如果调用涉及到修改操作，需要先在全网达成共识，之后修改操作会被记录在区块链，修改结果会被存在状态数据库(例如，转账交易的转账金额会被记录到区块链，账户余额的增减会被应用到状态数据库). 如果调用仅包含查询操作，则无需共识，也不需被记录在区块链上. 智能合约还支持合约内部事件的注册与通知机制，从而可主动向外部应用通知合约内部发生的关键事件. 智能合约目前只能访问链内数据，无法主动监听并响应链外事件.

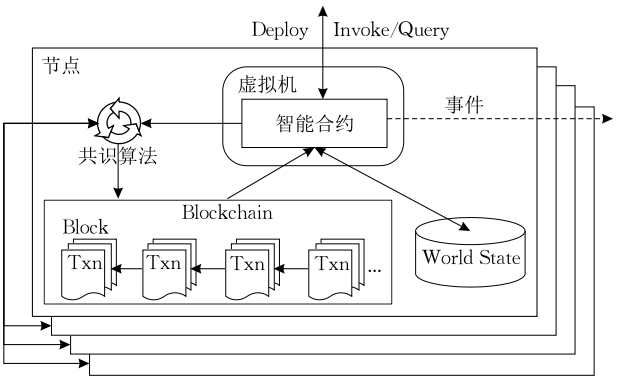

图 6　智能合约的运作机制

智能合约定义了交易逻辑及访问状态数据的业务规则，外部应用(如以太坊中的去中心化应用 Dapp)需要调用智能合约，并依照合约执行交易和访问状态数据. 外部应用与智能合约间的关系非常类似于传统数据库应用与存储过程间的关系，存储过程运行于数据库管理系统之中，访问关系数据库数据；而智能合约运行于区块链系统之中，访问区块和状态数据.

5.2　编程语言

比特币平台提供了处理交易的简单脚本，这些脚本是基于栈的一组指令，为了避免可能的漏洞与攻击，所以没有设计循环指令和系统函数. 因而比特币脚本不是图灵完备的程序语言，比特币平台不存在严格意义上的智能合约.

以太坊自定义了 Solidity、Serpent 等图灵完备的脚本语言以开发智能合约，自定义脚本语言是为了实现特殊的合约功能. 以太坊智能合约内置了表示“账户地址”的 address 数据类型，倾向于支持基于数字货币的支付应用. 以太坊的合约账户和外部账户共享同一地址空间，合约地址能被看做一个外部账户地址，可通过向合约地址发送交易来调用智能合约. 合约执行过程中依据占用的 CPU 和内存会消耗 Gas，Gas 由以太币兑换而来，一旦 Gas 耗尽，合约就会终止执行，消耗掉的费用不会退回，从而防范了垃圾交易或含有死循环的智能合约. Solidity、Serpent 等图灵完备的编程语言在增强合约逻辑功能、降低合约编写难度的同时，也会带来潜在的安全风险. 2016 年 6 月，以太坊上最大众筹项目 The DAO① 的智能合约因递归调用漏洞而遭遇攻击，约 1200 万个以太币被非法转移，后虽通过硬分叉追回了损失，但 The DAO 项目宣布失败并解散. 因此，Solidity 团队正在考虑整合形式化验证以保障智能合约代码的正确性.

Hyperledger Fabric 可基于 Go 和 Java 高级语言开发智能合约，这些高级语言不但图灵完备，编译技术成熟，而且也可减轻合约编程者的学习门槛. Hyperledger Fabric 的智能合约被称为 Chaincode，倾向于支持通用的企业级应用，其是与区块链交互的唯一渠道及生成交易的唯一来源. 编写合约实质是实现 Chaincode 接口中的 Init、Invoke 和 Query 三个函数，其分别用于实现状态数据的初始化、修改和查询.

5.3　沙箱环境

智能合约不能直接运行在区块链节点上，因为合约中若含有漏洞或恶意代码，就会直接威胁到区块链节点的安全，所以智能合约必须运行在隔离的沙箱环境中. 合约和宿主系统之间、合约与合约之间被沙箱执行环境有效隔离、互不干扰，这就限定了漏

① The DAO. https://en.wikipedia.org/wiki/The_DAO_(organization)

洞或恶意代码的影响范围.目前,主流区块链平台对沙箱的支持分为虚拟机和容器两类,它们的作用都是保证合约代码在沙箱中执行,以对合约使用的资源进行隔离和限制.

以太坊使用自定义的以太坊虚拟机(EVM)作为沙箱,运行由 Solidity、Serpent 等语言编译生成的字节码,这些字节码不能访问 EVM 宿主机的网络系统、文件系统和其它进程,合约之间也只有有限的调用. Hyperledger Fabric 使用轻量级的 Docker 容器作为沙箱,基于 Docker 自身提供的隔离性和安全性,保护了宿主机不受容器中恶意合约的攻击,也防止了容器之间的相互影响.

6 可扩展性

横向扩展性使得分布式数据库的整体性能随着集群节点数的增加而线性提升.比特币、以太坊和 Hyperledger Fabric 目前都采用全网节点共享一条区块链的单链方案,网络上的每个节点需要处理、存储全网的所有交易和全部数据,整个区块链系统的处理能力实际上受限于单个计算节点的处理能力.另外,受到共识算法的影响,随着节点数的增加,系统整体处理能力不但未随之提升,甚至还会降低.为了实现动态的可扩展性,以太坊应用了分片(sharding)[53]的解决方案,Hyperledger Fabric 应用了多通道(multichannel)①的解决方案.这些方案使得全网由原来的单链扩展到了多链(multi-chain),在多链上可同时并发的处理多笔交易,突破全网处理能力受限于单个节点的限制,从而提升系统整体性能.

6.1 分 片

为了解决以太坊系统吞吐量和存储容量的问题、支持全球范围的高频次交易,在 2016 年以太坊开发者大会(Devcon2)上,Buterin 发布了描述以太坊 2.0 的紫皮书[53],紫皮书中提出了分片处理交易的解决方案.以太坊依据账户地址将全网划分为多个相对独立的分片,每个分片内维护一条独立子链,用户可自行选择在哪个分片执行自己的交易,每个节点根据自身的计算和存储能力选择加入一到多个分片,并处理和存储这些分片上的交易.全网节点分工配合以覆盖到所有分片,如果需要访问本节点没有的交易数据,则利用轻客户端技术从其它分片节点读取.全网节点可并行的处理和存储不同的交易数据,使得全网交易处理能力不再受限于单一节点,单一节点也不需处理、存储全部数据.应用分片技术后,以太坊不再适宜使用 PoW 共识机制,因为分片会将全网算力分散,在单个分片内攻击者很容易突破 51%的算力,因此应用分片技术的以太坊 2.0 使用的是基于权益证明机制的 Casper 共识算法.

6.2 多通道

为了解决 Hyperledger Fabric 的系统扩展性和交易隐私性问题,Hyperledger Fabric 1.0 提出了多通道的方案.以太坊的分片技术从资源均衡的角度将整个区块链网络划分成为大小相同的多个分片;Hyperledger Fabric 的通道技术则基于交易规则将整个区块链网络划分为多个逻辑上的通道,每个节点根据自己需要参与的交易来选择加入相应的通道.如图 7 所示,节点 Peer1 加入到了通道 Ch1 和通道 Ch2,节点 Peer2 加入了通道 Ch2 和通道 Ch3,节点 Peer3 加入了通道 Ch1、通道 Ch2 和通道 Ch3.每个节点可以在多个链上同时接收和处理区块,多个链上的交易可以独立、并发地执行.相对于原来的单链结构,全网吞吐量将显著提升.Ordering 服务节点提供插件化的共识服务,可基于 Kafka 消息系统或 SBFT 共识对所有链上的每个交易进行统一排序,当其由可信方或监管机构组成时,就不涉及交易隐私泄露的问题;但若不希望 Ordering 服务节点获知交易的具体内容,则可利用加密来隐藏交易数据.

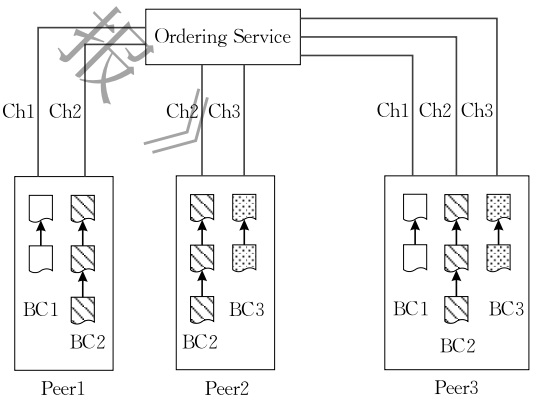

图 7 Hyperledger Fabric 多通道

在基于 PoW 共识机制的比特币和以太坊中,节点在任意时刻都可自由加入或退出. Hyperledger Fabric 采用的 PBFT 算法要求所有节点数目已知且静态不变,这就不利于区块链网络的动态扩展.为此,Hyperledger Fabric 1.0 将网络节点分为共识节点(orderer)和记账节点(committer),解耦了共识服

① Multichannel. https://hyperledgerdocs.readthedocs.io/en/latest/multichannel.html

务和记账服务，从而实现了记账节点的动态加入或退出.

7　安全性

比特币、以太坊等公有链允许任意节点加入网络、允许任何用户参与交易，全部账本数据公开透明且由多方共同维护. 这些有别于传统数据库的特性，使得区块链系统无法直接应用传统数据库的安全机制. 为了保障数据的安全性，传统数据库系统都设计了完善的用户管理和存取控制. 用户管理通常需要运行在中心化的节点上，这就和区块链所强调的去中心化相矛盾；另外，存取控制也与公有链所强调的数据公开性和透明性相矛盾. 目前，区块链系统主要基于数字签名与验证来确保数字货币的所有权以及交易的不可伪造、不可否认的特性；依靠每次交易使用不同的数字证书和账户地址来保障交易的隐私性.

7.1　签名与验证

基于公有链的比特币和以太坊没有提供用户管理服务，公钥是标识和区分一个用户的唯一方法. 为了保障交易安全性，需对每笔交易都进行签名与验证，签名算法使用了椭圆曲线数字签名算法(ECDSA). 图 8 反映了比特币交易的签名与验证，每当需要进行一笔转账交易时，接收者需要将其公钥的 SHA256 与 RIPEMD160 的双哈希结果 pubKeyHash，即比特币地址提供给发送者，pubKeyHash 会被放在交易的输出脚本 ScriptPubKey 中. 以 OP 开头的语句是比特币脚本中的操作指令；尖括号语句代表比特币脚本中的数据指令. 为了完成一笔交易，发送者需要对该笔交易数据进行签名，并把签名 sig 和其公钥 pubKey 放在输入脚本 ScriptSig 中. 签名 sig 表明比特币持有者本人承认该交易并亲自花出了前一笔交易中获得的比特币；公钥 pubKey 一方面用于验证签名 sig 的有效性，另一方面用于验证其哈希值是否和前一笔交易输出脚本中的比特币地址 pubKeyHash 一致，以保证发送者确实拥有这些比特币. 以上所有签名与验证过程都是基于输入脚本和输出脚本自动完成的.

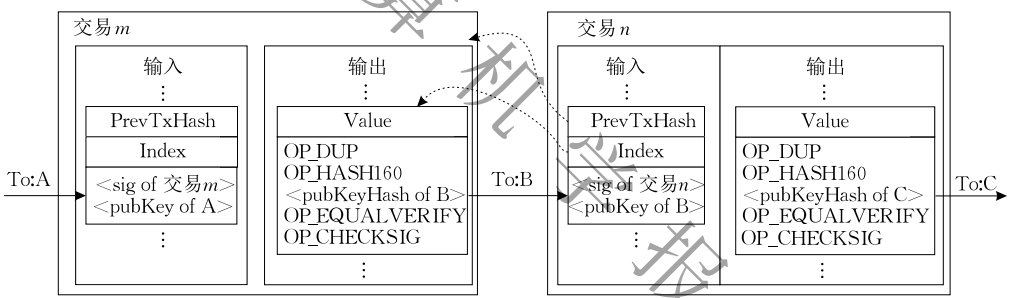

图 8　比特币交易的签名与验证

由于比特币是基于交易的模型，以太坊是基于账户的模型，所以它们对交易的签名与验证过程并不相同. 如图 9 所示，以太坊的交易数据中只包含了发送者的 ECDSA 签名，并不包含发送者的公钥和发送者的地址，因为基于 ECDSA 签名、原始交易数据和椭圆曲线参数可以恢复出发送者的公钥，然后对公钥进行 SHA3 哈希运算，即可计算出发送者的账户地址. 如此设计可减少每笔交易的字节数，从而减少交易数据在存储和网络方面的开销.

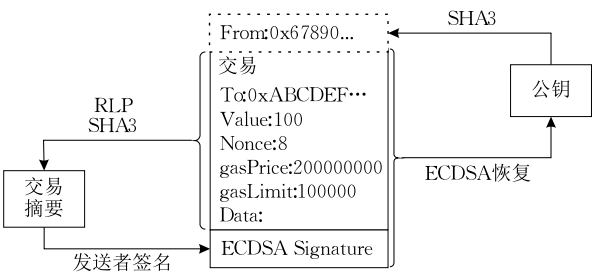

图 9　以太坊交易的签名与验证

Hyperledger Fabric 是针对联盟链而设计的，这就要求所有节点和用户必须经过认证和授权才可加入. Hyperledger Fabric 提供了专门的成员管理服务 Membership，并基于 CA 中心分别提供了 ECert(Enrollment Cert)、TCert(Transaction Cert)和 TLSCert(Transport Layer Security Cert)三种类型的数字证书，不同环节需要使用不同类型的证书. ECert 证书用于身份认证，在登录系统时可确认节点和用户的身份. TCert 证书用于交易的签名与验证，Hyperledger Fabric 的每笔交易都包含了发送者的签名和交易证书，为确保第三方无法由交易证书追溯出具体的发送者，每次交易可使用不同的 TCert 证书. TLSCert 证书用于系统组件间的 SSL/TLS 通讯.

7.2　隐私性

所有交易数据公开、透明地全量存储在全网每

个节点上，虽可防范数据伪造和篡改，但却带来了数据隐私问题. 比特币地址和以太坊账户地址都是公钥的哈希值，其与用户真实身份信息之间没有直接的对应关系，依靠隔断地址和用户真实身份之间的关联，实现了一定程度的匿名性. 另外，每次交易都使用全新的地址，不同地址之间没有任何关联，其实现了多个交易间的不可关联性. 但是，这些方案无法保障绝对的隐私性，例如，电子钱包或比特币交易所的实名认证、同笔交易中多个输入间的关联关系、网络报文中 IP 地址与比特币地址的对应关系等都可能泄露用户真实身份.

区块链隐私保护既需掩盖交易细节，又需验证交易的正确性. 区块链目前的隐私保护方案包括混币(CoinJoin)①、环签名(ring signature)[54]、零知识证明(Zero-Knowledge Proof)[55]、同态加密(homomorphic encryption)[56]等. 达世币(dash)应用了混币，将多笔交易拼凑成一笔交易一起执行，以打断发送地址和接收地址之间的联系. 门罗币(monero)应用了环形签名，用发送者的私钥和多个无关者的公钥加密交易数据，再用所有人的公钥解密数据，以此隐藏交易的发送者. Zerocash[57]应用了零知识证明，可在无需泄露交易数据、无任何额外信息的前提下，向验证者证明某个交易是正确的. Zerocash 实现了特定交易的隐私保护，通过扩展 Zerocash，Hawk[58]实现了具有隐私保护的智能合约，可支持任意交易的隐私保护. 同态加密基于同态映射保证了先运算后加密和先加密后运算的结果相同，从而可基于加密数据实现交易验证.

zkSNARKs[59]是应用于 Zerocash 的零知识证明实现方案，其可在无需执行且无需知道输入的前提下验证一个计算的正确性. Zerocash 使用 zkSNARKs 隐藏了交易关联性和转账金额，在无需泄露交易数据的前提下，完成交易的确认和验证. 以太坊正在扩展智能合约语言，计划在 EVM 中增加 SNAKE 验证指令以高效支持 zkSNARKs 证明的验证. 2016 年 7 月在康奈尔大学的 Ethereum/IC3 Bootcamp 上，研究者们在 Ethereum 上完成了 zkSNARKs 相关代码演示②.

Hyperledger Fabric 0.6 采用了单链的方案，每个用户可访问到链上的全部交易数据. 联盟链的用户主要是商业机构，交易数据是这些机构的重要资产和商业机密，因而隐私性就显得尤为重要. 为此，Hyperledger Fabric 1.0 采用了多通道的方案，任何需要交易的双方或多方可建立链接彼此的通道，其后所有交易都可在该通道上完成，通道上的所有交易数据会存储在单独的区块链上，只有通道上的用户可访问链上的数据，没有加入通道的用户将无权访问. 例如，图 7 中的节点 Peer1 只能参与和查看通道 Ch1 和通道 Ch2 上的交易，不能参与和查看通道 Ch3 上的交易. Hyperledger Fabric 1.0 的多通道将不同的交易分配到了相互隔离的多条区块链上，实现了私密的交易，保障了交易数据的隐私性.

8 区块链的优劣势与发展趋势

8.1 区块链的优势

区块链是一种多方共同维护的分布式数据库，与传统数据库系统相比，其主要优势如下：

(1) 去中心化. 传统数据库集中部署在同一集群内，由单一机构管理和维护. 区块链是去中心化的，不存在任何中心节点，由多方参与者共同管理和维护，每个参与者都可提供节点并存储链上的数据，从而实现了完全分布式的多方间信息共享.

(2) 不可篡改. 区块链依靠区块间的哈希指针和区块内的 Merkle 树实现了链上数据的不可篡改；而数据在每个节点的全量存储及运行于节点间的共识机制使得单一节点数据的非法篡改无法影响到全网的其它节点.

(3) 可追溯. 区块链上存储着自系统运行以来的所有交易数据，基于这些不可篡改的日志类型数据，可方便地还原、追溯出所有历史操作，其方便了监管机构的审计和监督工作.

(4) 高可信. 区块链是一个高可信的数据库，参与者无需相互信任、无需可信中介即可点对点的直接完成交易. 区块链的每笔交易操作都需发送者进行签名，必须经过全网达成共识之后，才被记录到区块链上. 交易一旦写入，任何人都不可篡改、不可否认.

(5) 高可用. 传统分布式数据库采用主备模式来保障系统高可用，主数据库运行在高配服务器上，备份数据库从主数据库不断同步数据；如果主数据库出现问题，备份数据库就及时切换作为主数据库. 这种架构方案配置复杂、维护繁琐且造价昂贵. 在区

① CoinJoin. https://bitcointalk.org/index.php?topic=279249.0
② zkSNARKs in Ethereum. https://z.cash/blog/zksnarks-in-ethereum.html

块链系统中，没有主备节点之分，任何节点都是一个异地多活节点. 少部分节点故障不会影响整个系统的正确运行，且故障修复后能自动从全网节点同步数据.

8.2 区块链的劣势

和发展了近 40 年的传统数据库相比，区块链尚处于技术发展的初期阶段，还有着很多不足需要克服：

(1) 吞吐量. 比特币和以太坊的吞吐量分别约为 7 TPS 和 25 TPS，Hyperledger Fabric 的吞吐量小于 2000 TPS，远低于现有的数据库. 传统数据库的每笔交易是被单独执行处理的，但区块链系统则以区块为单位攒够多笔交易再一批处理，这就延长了交易时间. 不论是基于 PoW 的公有链，还是基于 PBFT 的联盟链，其实质都是以牺牲性能来换取区块链系统的安全性，每笔交易的签名与验证、每个区块的哈希运算以及复杂的共识过程等都涉及大量的系统开销.

(2) 事务处理. 目前的区块链平台主要依赖底层数据库来提供事务处理，而底层数据库大多是没有事务处理能力的 Key-Value 数据库. 比特币、以太坊和 Hyperledger Fabric 都采用 LevelDB 存储区块链索引或状态数据，但 LevelDB 并不支持严格的事务. 单个节点上的智能合约执行失败会导致数据库数据不一致，必须从其它节点同步数据才能使本机数据恢复到一致状态.

(3) 并发处理. 传统数据库可高并发地为成百上千的客户端提供服务. 区块链的节点大多是以对等节点的身份参与 P2P 网络中的交易处理，并没有针对高并发服务做优化设计，因而无法支持高并发的客户端访问.

(4) 查询统计. 传统数据库提供了丰富的查询语句和统计函数. 区块链通常存储在 Key-Value 数据库甚至文件系统，在 Non-Key 查询和历史数据查询上都很不方便，更别说复杂的复合查询和统计. 区块链系统应实现插件化的数据访问机制，以支持包括关系数据库在内的多种数据库.

(5) 访问控制. 传统数据库具有成熟的访问控制机制. 目前大多数区块链平台的数据都是公开透明地全量存储在每个节点上，仅依靠交易的签名与验证，来确定资产的所有权和保证交易的不可伪造，除此之外，基本没有再提供其它的安全机制. 有别于传统数据库中心化的访问控制，如何针对区块链设计去中心化的访问控制也是函待解决的问题.

(6) 可扩展性. 传统数据库通过横向扩展增加节点数，以线性的提高系统吞吐量、并发访问量和存储容量. 目前大多数区块链平台随着节点数的增加其系统整体性能反而在下降，部分区块链平台提出的扩展性方案还需要时间的验证.

8.3 区块链的发展趋势

当应用于实际业务时，目前的区块链平台在诸多方面尚存在问题，为了解决这些问题，未来的区块链还需在以下几个方面进一步研究发展：

(1) 共识机制. 共识机制目前已经成为了区块链系统性能的关键瓶颈. 在基于证明机制的共识算法中，经受多年实践性安全检验的 PoW 机制有着消耗大量计算资源及性能低下的问题. 在基于投票机制的共识算法中，有着完善理论证明的 PBFT 算法面临着广播带来的网络开销过大的问题. 如何提高系统吞吐率是共识机制最迫切需要解决的问题，因此，包括在少部分可信节点中选取主节点的共识算法、保证高概率正确性的异步共识算法、基于特定安全性前提并减少网络广播的共识算法、基于可信硬件的共识算法、同时融合 PoW 与 PBFT 优势的共识算法在未来都是值得关注的.

(2) 隐私保护. 因为能够隐藏交易内容，零知识证明和同态加密是最受关注的隐私保护解决方案. 零知识证明目前更多被应用于数字货币领域，只有 Zerocash 和 Hawk 基于零知识证明构建了区块链应用和模型. 同态加密算法可抵抗量子计算的攻击，但其运算效率低，距离实际应用尚有较大差距. 因此，针对零知识证明、同态加密等隐私保护方案，如何扩大应用领域、提高运算效率、加快应用落地，将会是今后最迫切的研究工作.

(3) 部分存储. 比特币平台的每个网络节点都全量的存储着所有历史交易数据，这虽然保证了数据的公开性、透明性及系统的高可用性，但也带来数据隐私问题；另外，每个交易都需同步到全网所有节点，也会带来性能问题. 所以，很多平台采用了只存储部分交易数据的解决方案. Corda 主要应用于对数据隐私要求较高的金融领域，所以从一开始就反对区块链中每个节点存储全部数据，而使数据仅对交易双方及监管可见. Hyperledger Fabric 1.0 的多通道技术从性能和隐私两个角度考虑，使每个通道仅存储与通道节点有关的交易. 以太坊 2.0 的分片技术将全网交易数据按片数等分，使得每个分片存

储的交易数据尽可能均衡.随着交易量和数据量的剧增,区块链节点由全量存储到部分存储将会成为未来的一个趋势.

(4)链外交易.为了提高交易处理能力,比特币社区提出了增大区块、隔离见证(segregated witness)①和闪电网络(lightning network)[60]等扩容方案.当前比特币区块尺寸上限为1MB,比特币社区提出增大区块尺寸上限至2MB以容纳2倍的交易量.比特币交易的输入脚本包含有发送者的签名数据以证明其拥有该笔比特币,但签名数据仅仅用于矿工挖矿时做交易验证,没有其它额外的用途.隔离见证是将交易中的签名数据移出以减少交易尺寸使区块容纳更多交易.增大区块和隔离见证只是增加了区块容量,无法从根本上改善性能,但闪电网络可达到每秒百万级的交易量.闪电网络是一种提供比特币链外(off-chain)双向快速支付的通道,其提供了高频、小额、立即确认的支付方式,并且具有更好的隐私性和更低的手续费.雷电网络(Raiden network)②是根据闪电网络提出的以太坊链外快速支付通道.闪电网络和雷电网络把小额交易放在链外,既实现了高速交易,也减轻了主链压力,主链只处理最终的交易及作为争议仲裁的最后手段.闪电网络和雷电网络是目前提高交易处理能力最有效的方案,未来会有一定的发展空间.

(5)多链与侧链.传统区块链平台的单链设计方案使得系统整体处理能力受限于单个计算节点.多链设计方案可使互不相关的交易实现分片存储和并发执行,不但提高了系统的可扩展性,使全网不再受限于单个节点,而且链间隔离还保证了交易数据的隐私.除了以太坊中的分片、Hyperledger Fabric中的多通道,Monax、MultiChain③等区块链平台也提供了自己的多链方案.侧链(sidechain)最初是通过锚定比特币而实现数字资产交易的区块链技术,主要解决比特币平台应用单一、性能受限等问题.侧链是一个独立的区块链,有自己的账本、共识机制、交易类型和智能合约,通过锁定主链上的比特币,可使得相应数量的比特币在侧链上流通.例如,Blockstream④推出的元素链[61]通过与比特币双向锚定,既实现了比特币在主链和侧链间的互转,还提供了智能合约,私密交易等特性.通过为每个应用分别创建一个锚定到主链的侧链,可扩展传统区块链支持多种应用类型.多链与侧链能够解决现有区块链的问题和不足,未来需要进行更多研究.

(6)跨链.面对数量众多、类型各异的区块链平台,跨链技术可以实现它们之间的互联、互通及互信.以数字资产为例,如果能够打破不同区块链间的壁垒,即可实现各类数字资产的跨链交易,形成融合多种资产的价值互联网.目前较有影响力跨链技术是Polkadot[62]和Cosmos[63].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)的互连,每个平行链就是一个单独的区块链网络.Polkadot还以其它公有链为升级目标,最终让以太坊直接可与任何链进行通讯.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Inter-Blockchain Communication(IBC)协议实现不同Zone之间的互联.Cosmos专注于实现跨链的数字资产交易,而Polkadot则专注于实现通用的跨链通信.跨链技术目前还在研究和试验阶段,但如同TCP/IP在当今互联网的地位,未来非常需要对应的方案来实现区块链间的"万链互联".

(7)区块树和区块图.区块之间未必要由链表来组织,业界已提出用树和图来组织区块的方案[49,64].为了应对出块间隔时间减小带来的分叉问题,以太坊中引入GHOST协议,该协议承认叔块使得以太坊区块链实质上成为了树形结构.为了适应于物联网小额支付的场景,IOTA⑤区块链平台提出使用有向无循环图(DAG)来组织区块的方案Tangle[65],每块只包含一个交易且至少链接之前的两个区块以表示确认过两个交易,整个图根据结点权重计算最长链并作为主链.未来非常需要在区块树和区块图方面进行更多的研究与实践.

(8)SQL on Blockchain.随着区块链系统性能的改善及交易数据的积累,基于区块链的数据分析工作将会成为迫切需求.现有的技术人员更熟悉传统的关系数据库,现有的数据分析工具也基本都基于SQL构建,区块链中的区块数据、交易数据及状态数据更趋近于结构化数据.如同Hadoop⑥编程由MapReduce转向SQL on Hadoop、Spark⑦编程由RDD转向Spark SQL的发展历程一样,主流区

① Segregated witness. https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki
② Raiden network. https://raiden.network/
③ MultiChain. https://www.multichain.com
④ Blockstream. https://www.blockstream.com
⑤ IOTA. https://iota.org/
⑥ Hadoop. http://hadoop.apache.org/
⑦ Spark. http://saprk.apache.org/

块链平台在未来非常需要 SQL on Blockchain 的查询引擎，从而使现有技术人员能够快速上手，使现有数据分析工具能够无缝接入.

(9) BlockchainDB. 类似互联网企业的快速发展催生了一批优秀的 NoSQL 数据库，随着一批去中心、去中介的新互联网应用的出现，未来迫切需要一种从底层到上层都直接支持现有区块链特性的数据库，可称其为 BlockchainDB. 参照图 1 的层次结构可知，BlockchainDB 在各层的设计上可完全借鉴数据库领域已有的成果和技术. BlockchainDB 在网络层上应该是基于 P2P 协议的，便于实现各种节点的动态加入与退出，从底层网络协议支持去中心化的架构. 数据库领域在 P2P 数据库管理方面已有了多年的研究及实际产品，可借鉴其相关成果. BlockchainDB 在共识层上应该支持具有拜占庭容错的共识算法，为了支持公有链、联盟链的不同应用场景，其应该分别提供证明机制、投票机制的共识算法. 尽管数据库领域更多采用的是 Paxos、Raft 等 CFT 共识算法，但其设计经验仍然值得借鉴. 为了便于存储和检索，BlockchainDB 在数据层上可直接应用现有数据库的存储与索引技术来处理区块链中的状态数据与索引数据. 区块数据和传统数据库的预写式日志非常类似，它们都维护了所有的历史操作记录，都是在表数据之前写入，都是追加形式的写且支持数据重放，只不过预写日志不具备不可篡改性且不支持查询，但预写日志在高速写入等方面的研究可用于区块数据. 另外，区块中的交易数据具有可追溯的特性，但不论在基于交易还是基于账户的模型中，目前的追溯查询并不高效，因此可借鉴数据仓库和科学数据管理领域的数据溯源(data provenance)[66-68]来解决，数据溯源的查询表达具有严格的代数学基础[69-70]，且可在关系数据库上实现[71]. 在智能合约层，智能合约与当前数据库的存储过程类似，其响应外部事件的机制与触发器类似，因此可借鉴存储过程与触发器的设计经验，甚至可以实现类似 PL/SQL 或 TSQL 编写的智能合约. BlockchainDB 在应用层上应该原生支持 SQL，提供支持访问区块数据、交易数据、状态数据的 SQL 语句，使应用程序获得与访问传统数据库相同的接口，以降低应用开发人员、数据库管理员的学习门槛. 由于去中心化的区块链与中心化的传统数据库在体系上的差异，传统数据库相关技术并非可直接应用于 BlockchainDB，这就需要根据区块链的特性开展进一步研究.

9　结束语

在没有第三方权威机构的中介协调下，区块链在互不了解的交易双方间建立了可靠的信任，去中心化地实现了可信的价值传输，因此区块链被称为价值互联网或第二代互联网. 首先，区块链以较低的成本实现了点对点的价值传输，这会冲击到以银行为代表的传统金融机构；其次，区块链的去中心化特性消除了对第三方中介机构的需求，达成了对等的直接交易，实现了真正的共享经济，这将影响到以中介代理为核心业务的互联网公司[72]. 最后，挑战与机遇并存，区块链的发展同时会给云计算、大数据及物联网等行业的发展带来更多的想象空间. 所以区块链不仅仅是一种新型数据库，也是一场互联网价值革命，将会给众多行业带来深远影响.

参　考　文　献

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. White Paper, 2008

[2] Buterin V. A next-generation smart contract and decentralized application platform. White Paper, 2014

[3] Cachin C. Architecture of the hyperledger blockchain fabric//Proceedings of the Workshop on Distributed Cryptocurrencies and Consensus Ledgers (DCCL). Chicago, USA, 2016

[4] Brown R G, Carlyle J, Grigg I, et al. Corda: An introduction. White Paper, 2016

[5] Hearn M. Corda: A distributed ledger. White Paper, 2016

[6] McConaghy T, Marques R, Müller A, et al. BigchainDB: A scalable blockchain database. White Paper, 2016

[7] Beijing PeerSafe Technology Co., Ltd. White paper for blockchain database application platform. White Paper, 2017 (in Chinese)
(北京众享比特科技有限公司. 基于区块链的数据库应用平台技术白皮书. 白皮书, 2017)

[8] Tencent FiT, Tencent Research Institute. White paper for tencent trustSQL. White Paper, 2017(in Chinese)
(腾讯 FiT, 腾讯研究院. 腾讯区块链方案白皮书. 白皮书, 2017)

[9] Yuan Yong, Wang Fei-Yue. Blockchain: The state of the art and future trends. Acta Automatica Sinica, 2016, 42(4): 481-494(in Chinese)
(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)

[10] He Pu, Yu Ge, Zhang Yan-Feng, et al. Survey on blockchain technology and its application prospect. Computer Science, 2017, 44(4): 1-7(in Chinese)

(何蒲，于戈，张岩峰等. 区块链技术与应用前瞻综述. 计算机科学，2017，44(4)：1-7)

[11] Tschorsch F, Scheuermann B. Bitcoin and beyond: A technical survey on decentralized digital currencies. IEEE Communications Surveys and Tutorials, 2016, 18(3): 2084-2123

[12] Gribble S D, Halevy A Y, Ives Z G, et al. What can database do for peer-to-peer? //Proceedings of the Fourth International Workshop on the Web and Databases (WebDB). Santa Barbara, USA, 2001: 31-36

[13] Yu Min, Li Zhan-Huai, Zhang Long-Bo. P2P data management. Journal of Software, 2006, 17(8): 1717-1730(in Chinese)
(余敏，李战怀，张龙波. P2P 数据管理. 软件学报，2006，17(8)：1717-1730)

[14] Qian Wei-Ning. Data Management in Peer-to-Peer Systems [Ph. D. dissertation]. Fudan University, Shanghai, 2004(in Chinese)
(钱卫宁. 对等计算系统中的数据管理[博士学位论文]. 复旦大学，上海，2004)

[15] Antonopoulos A M. Mastering Bitcoin: Unlocking Digital Cryptocurrencies. Sebastopol, USA: O'Reilly Media, Inc., 2014

[16] Lamport L. The part-time parliament. ACM Transactions on Computer Systems, 1998, 16(2): 133-169

[17] Lamport L. Paxos made simple. ACM Sigact News, 2001, 32(4): 18-25

[18] Ongaro D, Ousterhout J K. In search of an understandable consensus algorithm//Proceedings of the USENIX Annual Technical Conference. Philadelphia, USA, 2014: 305-319

[19] Lamport L, Shostak R, Pease M. The Byzantine generals problem. ACM Transactions on Programming Languages and Systems, 1982, 4(3): 382-401

[20] Fischer M J, Lynch N A, Paterson M S. Impossibility of distributed consensus with one faulty process. Journal of the ACM, 1985, 32(2): 374-382

[21] Castro M, Liskov B. Practical Byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems, 2002, 20(4): 398-461

[22] Kotla R, Alvisi L, Dahlin M, et al. Zyzzyva: Speculative Byzantine fault tolerance//Proceedings of the 21st ACM Symposium on Operating Systems Principles 2007 (SOSP). Stevenson, USA, 2007: 45-58

[23] Kwon J. Tendermint: Consensus without mining. White Paper, 2014

[24] Liu S, Viotti P, Cachin C, et al. XFT: Practical fault tolerance beyond crashes//Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, USA, 2016: 485-500

[25] Behl J, Distler T, Kapitza R. Scalable BFT for multi-cores: Actor-based decomposition and consensus-oriented parallelization //Proceedings of the 10th Workshop on Hot Topics in System Dependability (HotDep). Broomfield, USA, 2014: 9-14

[26] Zbierski M. Parallel Byzantine fault tolerance//Wilinski A, Fray I, Pejas J, eds. Soft Computing in Computer and Information Science. Switzerland: Springer International Publishing, 2015: 321-333

[27] Zhao W. Optimistic Byzantine fault tolerance. International Journal of Parallel, Emergent and Distributed Systems, 2016, 31(3): 254-267

[28] Schwartz D, Youngs N, Britto A. The ripple protocol consensus algorithm. White Paper, 2014

[29] Douceur J R. The sybil attack//Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS). Cambridge, USA, 2002: 251-260

[30] Dwork C, Naor M. Pricing via processing or combatting junk mail//Proceedings of the Advances in Cryptology-CRYPTO'92 (CRYPTO). Santa Barbara, USA, 1992: 139-147

[31] Aspnes J, Jackson C, Krishnamurthy A. Exposing computationally-challenged Byzantine impostors. Yale University, New Haven, USA: Technical Report YALEU/DCS/TR-1332, 2005

[32] King S, Nadal S. PPCoin: Peer-to-peer crypto-currency with proof-of-stake. White Paper, 2012

[33] Larimer D. Delegated proof-of-stake. White Paper, 2014

[34] Bayer D, Haber S, Stornetta W S. Improving the efficiency and reliability of digital time-stamping//Capocelli R, et al, eds. Sequences II: Methods in Communication, Security and Computer Science. New York, USA: Springer-Verlag, 1993: 329-334

[35] Haber S, Stornetta W S. How to time-stamp a digital document //Proceedings of the Advances in Cryptology-CRYPTO'90 (CRYPTO). Santa Barbara, USA, 1990: 437-455

[36] Haber S, Stornetta W S. Secure names for bit-strings//Proceedings of the 4th ACM Conference on Computer and Communications Security (CCS). Zurich, Switzerland, 1997: 28-35

[37] Merkle R C. Protocols for public key cryptosystems//Proceedings of the 1980 IEEE Symposium on Security and Privacy (S&P). Oakland, USA, 1980: 122-134

[38] Merkle R C. A digital signature based on a conventional encryption function//Proceedings of the Advances in Cryptology-CRYPTO'87 (CRYPTO). Santa Barbara, USA, 1987: 369-378

[39] Szydlo M. Merkle tree traversal in log space and time//Proceedings of the Advances in Cryptology-EUROCRYPT 2004 (EUROCRYPT). Interlaken, Switzerland, 2004: 541-554

[40] Narayanan A, Bonneau J, Felten E, et al. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton: Princeton University Press, 2016

[41] Szabo N. Formalizing and securing relationships on public networks. First Monday, 1997, 2(9)

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

[43] Shentu Qing-Chun. Development Guide of Blockchain. Beijing: China Machine Press, 2017(in Chinese)
(申屠青春. 区块链开发指南. 北京：机械工业出版社，2017)

[44] Yang Bao-Hua, Chen Chang. Principle, Programming and Applications of Blockchain. Beijing: China Machine Press, 2017(in Chinese)
(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)

[45] Morrison D R. PATRICIA—Practical algorithm to retrieve information coded in alphanumeric. Journal of the ACM, 1968, 15(4): 514-534

[46] Hagmann R. Reimplementing the cedar file system using logging and group commit//Proceedings of the 11th ACM Symposium on Operating System Principles(SOSP). Austin, USA, 1987: 155-162

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

[48] Wattenhofer R. The Science of the Blockchain. Charleston, USA: CreateSpace Independent Publishing Platform, 2016

[49] Sompolinsky Y, Zohar A. Accelerating bitcoin's transaction processing. Fast money grows on trees, not chains. IACR Cryptology ePrint Archive, 2013

[50] Dinh T T A, Wang J, Chen G, et al. BLOCKBENCH: A framework for analyzing private blockchains//Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD). Chicago, USA, 2017: 1085-1100

[51] Bessani A, Sousa J, Alchieri E E P. State machine replication for the masses with BFT-SMART//Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). Atlanta, USA, 2014: 355-362

[52] Miller A, Xia Y, Croman K, et al. The honey badger of BFT protocols//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS). Vienna, Austria, 2016: 31-42

[53] Buterin V. Ethereum 2.0 mauve paper. White Paper. 2016

[54] Rivest R, Shamir A, Tauman Y. How to leak a secret//Proceedings of the Advances in Cryptology-ASIACRYPT 2001(ASIACRYPT). Gold Coast, Australia, 2001: 552-565

[55] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems. SIAM Journal on Computing, 1989, 18(1): 186-208

[56] Gentry C. Fully homomorphic encryption using ideal lattices //Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC). Bethesda, USA, 2009: 169-178

[57] Sasson E B, Chiesa A, Garman C, et al. Zerocash: Decentralized anonymous payments from Bitcoin//Proceedings of the IEEE Symposium on Security and Privacy (S&P). Berkeley, USA, 2014: 459-474

[58] Kosba A, Miller A, Shi E, et al. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts// Proceedings of the IEEE Symposium on Security and Privacy (S&P). San Jose, USA, 2016: 839-858

[59] Ben-Sasson E, Chiesa A, Genkin D, et al. SNARKs for C: Verifying program executions succinctly and in zero knowledge// Proceedings of the Advances in Cryptology-CRYPTO 2013 (CRYPTO). Santa Barbara, USA, 2013: 90-108

[60] Poon J, Dryja T. The Bitcoin lightning network: Scalable off-chain instant payments. White Paper, 2015

[61] Back A, Corallo M, Dashjr L, et al. Enabling blockchain innovations with pegged sidechains. White Paper, 2014

[62] Wood G. Polkadot: Vision for a heterogeneous multi-chain framework. White Paper, 2016

[63] Kwon J, Buchman E. Cosmos: A network of distributed ledgers. White Paper, 2016

[64] Lewenberg Y, Sompolinsky Y, Zohar A. Inclusive block chain protocols//Proceedings of the Financial Cryptography and Data Security. San Juan, Puerto Rico, 2015: 528-547

[65] Popov S. The tangle. White Paper, 2016

[66] Cheney J, Chiticariu L, Tan W C. Provenance in databases: Why, how, and where. Foundations and Trends in Databases, 2009, 1(4): 379-474

[67] Buneman P, Khanna S, Tan W C. Why and where: A characterization of data provenance//Proceedings of the Database Theory-ICDT 2001(ICDT). London, UK, 2001: 316-330

[68] Dong X L, Tan W C. A time machine for information: Looking back to look forward. Proceedings of the VLDB Endowment, 2015, 8(12): 2044-2045

[69] Green T J, Karvounarakis G, Tannen V. Provenance semirings //Proceedings of the 26th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS). Beijing, China, 2007: 31-40

[70] Green T J, Tannen V. The semiring framework for database provenance//Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS). Chicago, USA, 2017: 93-99

[71] Srivastava D, Velegrakis Y. Using queries to associate metadata with data//Proceedings of the 23rd International Conference on Data Engineering (ICDE). Istanbul, Turkey, 2007: 1451-1453

[72] Zhou Ao-Ying. Understanding on the big data: Beyond the data management and analytics. Big Data Research, 2017, 3(2): 3-18(in Chinese)
(周傲英. 感悟大数据——从数据管理和分析说起. 大数据，2017，3(2)：3-18)

SHAO Qi-Feng, born in 1976, M.S., associate professor. His research interests include big data and blockchain.

JIN Che-Qing, born in 1977, Ph. D., professor, Ph. D. supervisor. His research interests include location-based services, data stream management, uncertain data management, data quality, and database benchmark.

ZHANG Zhao, born in 1977, Ph. D., associate professor. Her research interests include massive data management and data mining.

QIAN Wei-Ning, born in 1976, Ph. D., professor, Ph. D. supervisor. His research interests include Web data management and social analysis and mining.

ZHOU Ao-Ying, born in 1965, Ph. D., professor, Ph. D. supervisor. His research interests focus on data management and applications, inclusive of Web data management, data intensive computing, in-memory cluster computing, big data benchmarking and performance optimization.

Background

This paper surveys recent research work on blockchain technology that belongs to the blockchain category. In legacy business networks, all participants maintain their own databases with duplication and discrepancies that result in disputes, increased settlement times, and the need for intermediaries with their associated overhead costs. However, by using blockchain-based shared databases, where transactions cannot be altered once validated by consensus and written to the databases, businesses can save time and costs while reducing risks.

Blockchain is a decentralized, trustless, tamper-proof and traceable distributed database maintained by multiple participants in a public or permissioned peer-to-peer network. Bitcoin is the most successful blockchain application, Ethereum is the first for to introduce smart contracts in the blockchain and the most popular public blockchain platform, and hyperledger fabric is the most popular enterprise blockchain platform. The existing papers mainly discussed the blockchain based on Bitcoin, this paper proposes an architecture model of the blockchain system based on three mainstream blockchain platforms (Bitcoin, Ethereum, Hyperledger Fabric), and discusses the principles and technologies of blockchain according to blockchain data, consensus mechanism, smart contract, scalability and security. This paper also analyzes the advantages, disadvantages and technology trends of blockchain by comparing with traditional databases.