

文章编号: 1004-4353(2021)03-0233-05

区块链技术中高效散列函数的设计与实现

滕飞, 李永珍*

(延边大学 工学院, 吉林 延吉 133002)

摘要: 为提高传统的 SHA-256 算法在区块链技术中的计算效率,对传统的 SHA-256 算法中的迭代结构进行了改进. 实验结果表明,在保证数据的安全性下,改进的 64 次迭代的 SHA-256 算法和改进的 32 次迭代的 SHA-256 算法的计算效率比传统的 SHA-256 算法分别提高了 24% 和 140%,改进的 32 次迭代的 SHA-256 算法的平均计算效率比 MD5 算法提高了 32%. 因此,该改进算法可为提高区块链技术的计算效率提供参考.

关键词: 区块链; 哈希函数; SHA-256 算法

中图分类号: TP391.41

文献标识码: A

Design and implementation of efficient hash function in blockchain technology

TENG Fei, LI Yongzhen*

(College of Engineering, Yanbian University, Yanji 133002, China)

Abstract: In order to solve the problem that the traditional SHA-256 algorithm has relatively low computational efficiency in blockchain technology, the iterative structure of the traditional SHA-256 algorithm is improved. Experimental results show that the computing efficiency of the improved SHA-256 algorithm with 64 iterations and 32 iterations is 24% and 140% higher than that of the traditional SHA-256 algorithm, respectively, and the SHA-256 algorithm with 32 iterations is 32% higher than that of MD5 algorithm on average, while the security of the improved SHA-256 algorithm is guaranteed. Therefore, the improved algorithm can provide a reference for improving the computational efficiency of blockchain technology.

Keywords: blockchain; Hash function; SHA-256 algorithm

0 引言

区块链技术由于具有能保障数据安全、提高协同效率和控制风险等优点,因此其在数字货币、智能合约、供应链管理、物联网金融等方面得到应用^[1-2]. 但随着区块链技术的广泛应用,现有的区块链技术在处理数据的效率方面逐渐难以满足人们的要求,因此提高区块链技术处理数据的效能具有重要意义. 针对区块链中的哈希函数计算效率较低进而影响区块链整体计算效率的问

题,文献[3]提出了一种基于 FPGA 的区块链哈希加速优化算法,但文献的作者并未对该优化算法的安全性方面进行说明. 文献[4]对 SHA-2 系列算法进行了改进,改进后的算法的安全性虽然比原算法有所提高,但其效率与原算法基本相同. 基于上述研究,本文针对 SHA-256 的迭代结构进行改进,并通过效率测试实验和雪崩效应实验验证了改进后的 SHA-256 算法可有效提高区块链的效率和安全性.

收稿日期: 2021-03-17

* 通信作者: 李永珍(1971—),男,博士,副教授,研究方向为网络安全和无线网络协议.

1 相关技术简介

1.1 区块链中的数据结构

区块链中的数据结构通常由区块头(块头)和区块体(块身)组成,如图 1 所示^[5]. 区块头一般包括版本号、前一区块的 Hash 值(Hash 指针)、随机数、目标 Hash 值(本区块的 Hash 值)、Merkle 根等. 区块体保存的是若干条记录以及由每条记录的 Hash 值构成的二叉 Merkle 树^[6]. 由于哈希函数具有单向性、抗碰撞性等特点,因此其目前被广泛应用于区块链技术中. 哈希算法在区块链技术中的主要作用是:将一个交易区块中的交易信息转换成一个哈希值(区块链通过比对交易信息的哈希值来确定信息有无被篡改),以及用于数据加密、共识计算的工作量证明、区块之间的链接等^[7-9].

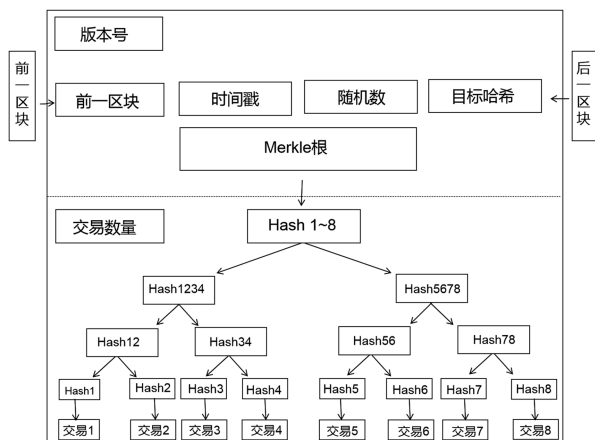


图 1 区块链数据结构图

1.2 SHA-256 算法

传统的 SHA-256 算法是将任意长度的字符串压缩成固定长度的字符串的一种算法. 该算法的压缩过程为:首先利用一种不可逆的方式将接收到的一段明文转化为一段长度较短、位数固定的散列值,然后通过处理明文生成一个 256 bit 长的哈希值(消息摘要),最后通过比对哈希值确定信息是否被篡改^[10].

SHA-256 算法的加密过程如下:

第 1 步 初始化常量. 初始化常量包括 8 个哈希初值(如表 1 所示)和 64 个哈希常量.

第 2 步 信息处理. 信息处理的方式是在报文末尾对数据进行填充. 填充方法是:补充第 1 个

比特,为数字 1;剩余的比特数补充数字 0. 填充的结果是报文长度对 512 取模以后的余数为 448.

第 3 步 计算消息摘要. 本文采用函数迭代的方式计算消息摘要,由此共生成 64 个字. 其中前 16 个字(记为 $w[0], \dots, w[15]$)由块分解(将消息分解成 512 bit 大小的块,每个字的大小为 32 bit^[11])产生,其余 48 个字由迭代公式得到. 迭代公式^[12]为:

$$W_t = \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}.$$

表 1 SHA-256 算法的哈希初值

h0	h1	h2	h3
0x6a09e667	0xbb67ae85	0x3c6ef372	0xa54ff53a
h4	h5	h6	h7
0x510e527f	0x9b05688c	0x1f83d9ab	0x5be0cd19

SHA-256 算法中的映射函数一共包含 64 次加密循环次数,如图 2 所示. 在图 2 中:ABCDEFGH 这 8 个字(word)按照一定的规则进行更新,其初始值分别为 $h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7$; K_i 对应的是 64 个常量; W_t 是本区块产生的第 t 个字. 对 ABCDEFGH 这 8 个字进行循环加密的过程中,最后一次循环所产生的 8 个字合起来即是第 i 个块对应到的散列字符串. SHA-256 算法中的逻辑运算如表 2 所示.

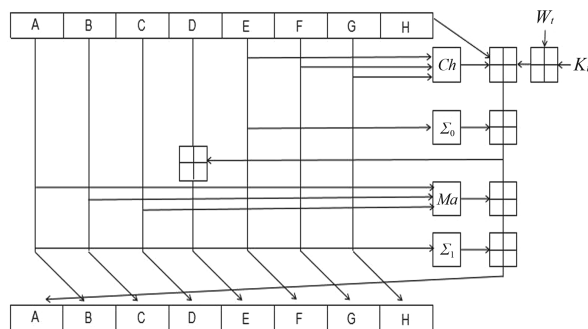


图 2 映射函数的加密循环示意图

表 2 SHA-256 算法中的逻辑运算

函数	表达式
$Ch(x, y, z)$	$(x \wedge y) \oplus (\neg x \wedge z)$
$Ma(x, y, z)$	$(x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$
$\Sigma_0(x)$	$S^2(x) \oplus S^{13}(x) \oplus S^{22}(x)$
$\Sigma_1(x)$	$S^6(x) \oplus S^{11}(x) \oplus S^{25}(x)$
$\sigma_0(x)$	$S^7(x) \oplus S^{18}(x) \oplus R^3(x)$
$\sigma_1(x)$	$S^{17}(x) \oplus S^{19}(x) \oplus R^{10}(x)$

2 改进的 SHA-256 算法

针对传统的 SHA-256 算法无法满足区块链对大数据的快速处理,本文对传统的 SHA-256 算法做如下改进:首先将 SHA-256 算法按 512 位进行分组划分,并且将每一个分组再划分为 8 个 64 位子分组;其次对报文进行处理后,算法的输出由原来的 8 个 32 位分组变为 4 个 64 位分组;最后将上述 4 个 64 位分组建联生成一个 256 位的散列值。

改进的 SHA-256 算法的执行过程如下:

第 1 步 在报文末尾处进行数据填充,填充后的报文长度对 512 取模后其余数为 448. 填充的方

法是首先在原报文末尾处添加 1 个比特(用数字 1 表示),然后补充剩余的比特数(用数字 0 表示)。

第 2 步 用 64 位比特数记录填充前的信息长度。

第 3 步 在改进的 SHA-256 算法中装入标准的幻数,标准的幻数为:

- A 为 0x36d219f41e0342b5,
- B 为 0xf807493c33fac611,
- C 为 0x1479c317abbfce09,
- D 为 0x4dbabfa350149e5c.

第 4 步 对改进的 SHA-256 算法进行 4 轮循环运算(每轮 16 个步骤),各轮所使用的线性函数如表 3。

表 3 每轮循环运算中的线性函数

轮数	参数取值范围	函数定义
1	$0 \leq t \leq 15$	$FF(a,b,c,d,M_j,s,t_i): a = b + ((a + F(b,c,d) + M_j + t_i) \lll s)$ $F(X,Y,Z) = (X \& Y) \mid ((\sim X) \& Z)$
2	$16 \leq t \leq 31$	$GG(a,b,c,d,M_j,s,t_i): a = b + ((a + G(b,c,d) + M_j + t_i) \lll s)$ $G(X,Y,Z) = (X \& Z) \mid (Y \& (\sim Z))$
3	$32 \leq t \leq 47$	$HH(a,b,c,d,M_j,s,t_i): a = b + ((a + H(b,c,d) + M_j + t_i) \lll s)$ $H(X,Y,Z) = X \wedge Y \wedge Z$
4	$48 \leq t \leq 63$	$II(a,b,c,d,M_j,s,t_i): a = b + ((a + I(b,c,d) + M_j + t_i) \lll s)$ $I(X,Y,Z) = Y \wedge (X \mid (\sim Z))$

第 5 步 主循环操作.主循环操作共有 4 轮,每轮有 16 次操作,共 64 次迭代次数.每次操作的单次循环过程如图 3 所示.每次操作时,首先将 4 个链接变量 A、B、C、D 赋值给变量 a、b、c、d;然后将赋值后的变量(a、b、c、d)和变量 s 以及常量 M_j (M_j 表示消息的第 j ($j=0 \sim 7$) 个子分组,其顺序如表 4 所示)、 t_i ($t_i=2^{64} * |\sin(i)|$) 代入非线性函数进行运算,并用运算得到的结果对应地取代 a、b、c、d.改进的 SHA-256 算法的运算流程如图 4 所示。

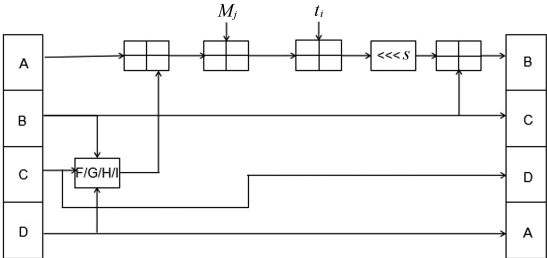


图 3 改进的 SHA-256 算法中的单次循环示意图

表 4 名文字顺序

参数 t 的取值范围	明文字的顺序
0~15	0,1,2,3,4,5,6,7,7,2,4,6,1,5,0,3
16~31	2,7,5,6,0,3,1,4,5,7,2,0,1,3,4,6
32~47	7,1,3,4,0,5,6,2,4,0,2,3,6,5,1,7
48~63	2,7,3,4,5,0,1,6,0,2,3,4,1,7,5,6

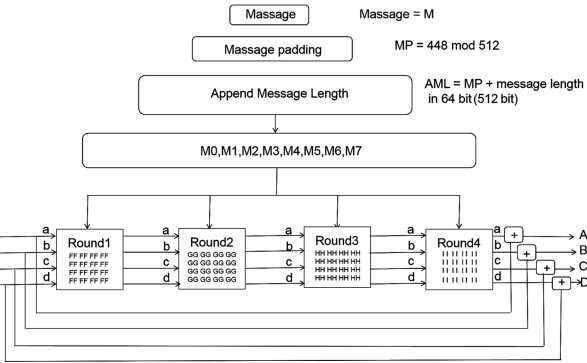


图 4 改进的 SHA-256 算法的运算流程

为了降低传统 SHA-256 算法的复杂度,对上述改进的 SHA-256 算法进行优化,即将算法

中主循环的每轮 16 次迭代变为 8 次迭代,由此 4 轮迭代可共减少 32 次迭代. 减少迭代次数(32 次)后,明文字 M_j 的数量和顺序如表 5 所示. 对比表 4 可知,算法减少 32 次迭代后,每轮所需要的明文字数量比之前减少了一半,由此表明减少迭代次数可有效提高算法的效率.

表 5 明文字顺序

参数 t 的取值范围	明文字的顺序
0~15	0, 1, 2, 3, 4, 5, 6, 7
16~31	2, 7, 5, 6, 0, 3, 1, 4
32~47	7, 1, 3, 4, 0, 5, 6, 2
48~63	2, 7, 3, 4, 5, 0, 1, 6

3 安全性分析

1) 穷举攻击. 本文采用天河二号超级计算机(计算速度为每秒 3.39×10^{16} 次)对改进的 SHA-256 算法进行抗穷举攻击验证. 经计算知,采用天河二号超级计算机进行穷举攻击破解时平均需要运行 1.22×10^{53} a,由此说明改进的 SHA-256 算法对抗穷举攻击具有良好的安全性.

2) 生日攻击. 由生日攻击原理可知,对改进后的 SHA-256 算法进行生日攻击时大概需要尝试 2^{128} 次哈希,若使用天河二号超级计算机对改进后的 SHA-256 算法进行生日攻击破解时平均需要运行 3.6×10^{14} a,这说明改进后的 SHA-256 算法对生日攻击具有良好的安全性.

3) 差分攻击. 由于改进的 SHA-256 算法与传统的 SHA-256 算法具有相同的迭代次数(64 次),因此改进的 SHA-256 算法也可以有效抵抗现有的差分攻击.

4) 雪崩效应. 选取 100 对报文进行雪崩效应实验. 改进后的 SHA-256 算法和传统的 SHA-256 算法的雪崩效应实验结果见图 5—图 7. 由图可以看出,改进的 64 次迭代的 SHA-256 算法和改进的 32 次迭代的 SHA-256 算法的坏点出现概率分别为 3% 和 5%,而传统的 SHA-256 算法的坏点出现概率为 5%. 这表明改进的 64 次迭代的 SHA-256 算法的雪崩效果优于传统的 SHA-256 算法.

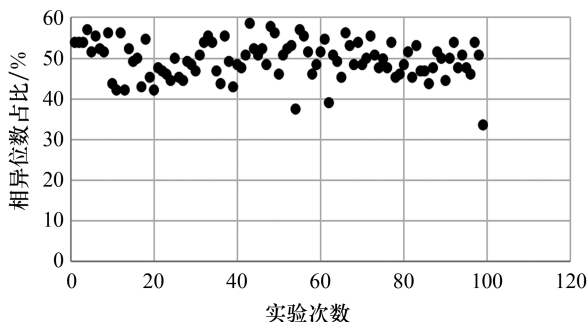


图 5 改进后的 SHA-256(32 steps)算法的雪崩实验结果

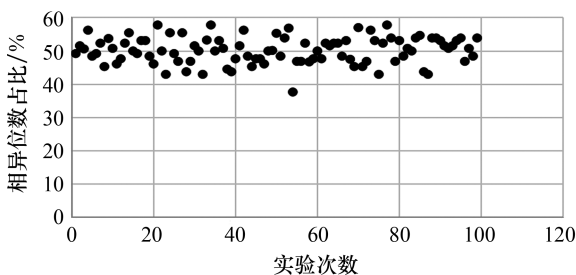


图 6 改进后的 SHA-256(64 steps)算法的雪崩实验结果

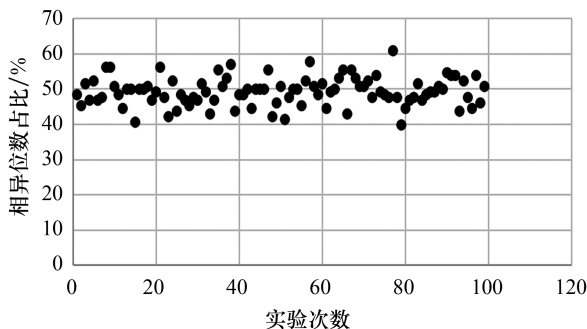


图 7 传统的 SHA-256(64 steps)算法的雪崩实验结果

4 性能分析

为了验证方法的有效性,将改进的 SHA-256 算法的处理速率分别与传统的 SHA-256 算法和 MD5 算法的处理速率进行了比较. 实验利用 Python 语言编程实现. 计算机的配置为: Interl(R) Core(TM) i5-5200U CPU@2.2 GHZ 处理器, 64 位 win7 操作系统. 实验数据选取 6 个不同字节的数据,加密次数为 10 000 次. 实验结果如图 8—图 10 所示. 由图可以看出,改进的 64 次迭代的 SHA-256 算法和 32 次迭代的 SHA-256 算法的计算效率比传统的 SHA-256 算法平均分别提高了 24% 和 140%,改进的 32 次迭代的 SHA-256 算法平均计算效率比 MD5 算法提高了 32%.

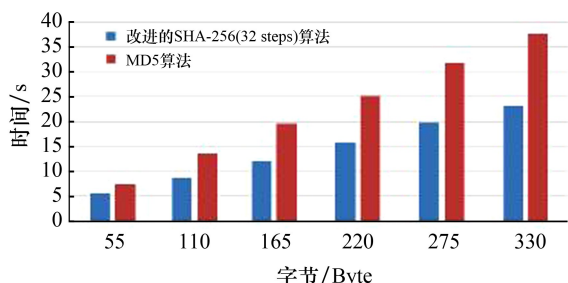


图8 改进的SHA-256(32 steps)算法和MD5算法的处理速率

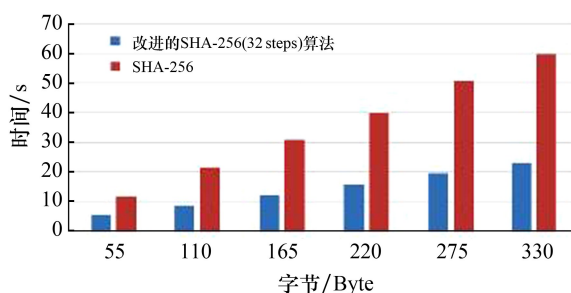


图9 改进的SHA-256(32 steps)算法和传统的SHA-256算法的处理速率

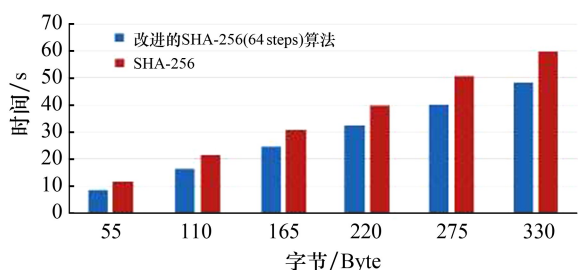


图10 改进的SHA-256(64 steps)算法和传统的SHA-256算法的处理速率

5 结论

研究表明,在保证算法安全性的基础上,本文改进的SHA-256算法(64 steps和32 steps)的计算效率比传统的SHA-256算法分别提高了24%和140%,改进的32次迭代的SHA-256算法的平均计算效率比MD5算法提高了32%,因此本文研究方法可为区块链技术提高数据处理效率提供参考。在今后的研究中,我们将在哈希函数中引

入随机组合结构,以此进一步提高本文改进算法的安全性和适用性。

参考文献:

- [1] 周李京. 区块链隐私关键技术研究[D]. 北京:北京邮电大学,2019.
- [2] 王化群,吴涛. 区块链中的密码学技术[J]. 南京邮电大学学报(自然科学版),2017,37(6):61-67.
- [3] 付金华. 高效能区块链关键技术及应用研究[D]. 郑州:战略支援部队信息工程大学,2020.
- [4] 刘飞. Hash函数研究与设计[D]. 南京:南京航空航天大学,2012.
- [5] LIU Y H, ZHANG S. Information security and storage of Internet of Things based on block chains[J]. Future Gener Comp Sy, 2020,106:296-303.
- [6] 韩璇,袁勇,王飞跃. 区块链安全问题:研究现状与展望[J]. 自动化学报,2019,45(1):206-225.
- [7] LI W T, LU Z P. Research on supply chain management based on block chain technology[C]//Proceedings of the 2nd Asia-Pacific Social Science and Modern Education Conference (SSME 2019). Shanghai: Atlantis Press, 2019:7-11.
- [8] PEREZ M R, GERARDO B, Medina R. Applying modified SHA256 on blockchain using challenge response and off-chain signatures patterns[C]//Proceedings of the the 3rd International Conference on Graphics and Signal Processing (ICGSP 2019). Visayas: ACM, 2019:58-61.
- [9] 陈震宇. 面向联盟区块链的高性能关键技术研究[D]. 杭州:浙江工业大学,2020.
- [10] ALGREDO-BADILLO I, FERREGRINO-URIBE C, CUMPLIDO R, et al. FPGA-based implementation alternatives for the inner loop of the Secure Hash Algorithm SHA-256[J]. Microprocess Microsy, 2013,37(6/7):750-757.
- [11] MICHAIL H E, ATHANASIOU G S, KELEFOURAS V, et al. On the exploitation of a high-throughput SHA-256 FPGA design for HMAC[J]. ACM T Reconfig Techn(TRETS), 2012,5(1):1-28.
- [12] HUANG X Y, SHEN F, TENG J H. The nonlinear analysis of SHA-256[C]//Proceedings of the 2011 IEEE International Conference on Information Theory and Information Security (ICITIS 2011). Zhengzhou: IEEE, 2011:88-90.