·安全技术·　　文章编号：1000-3428(2015)01-0130-05　　文献标识码：A　　中图分类号：TP309

一种交叉处理的混沌多变量 Hash 算法构造

张文婷，龙　敏

（长沙理工大学计算机与通信工程学院，长沙 410014）

摘　要：在现有的并行处理模式下，Hash 函数由于明文分块之间关联性不大从而引起安全问题。为此，提出一种交叉处理的多变量混沌 Hash 算法，算法安全性基于二次多变量多项式方程组求解问题（MQ 问题）的困难性和混沌理论的复杂性。其中 64 个压缩函数可并行处理数据，利用多变量代数理论构造输出函数进一步混乱与扩散，根据不同的需求调整 Hash 值的长度。对算法分别进行存储空间分析、伪造攻击分析、差分攻击分析及统计实验分析，结果表明，该算法弥补了传统多变量多项式密码的运行效率不足，且可以抵抗伪造攻击、差分攻击和统计攻击。

关键词：Hash 函数；MQ 问题；混沌映射；交叉处理；并行模式

中文引用格式：张文婷，龙　敏. 一种交叉处理的混沌多变量 Hash 算法构造[J]. 计算机工程，2015，41(1)：130-134.

英文引用格式：Zhang Wenting，Long Min. Chaos Multivariate Hash Algorithm Construction of Cross Processing[J]. Computer Engineering，2015，41(1)：130-134.

Chaos Multivariate Hash Algorithm Construction of Cross Processing

ZHANG Wenting，LONG Min

（College of Computer and Communication Engineering，Changsha University of Science and Technology，Changsha 410014，China）

【Abstract】Aiming at the defects of security in the existed parallel Hash funtions which are caused by the weak correlations between the plaintext block，a novel Hash function algorithm based on the difficulty of solving MQ problem and the complexity of chaotic theory is proposed. The algorithm works in a parallel and cross processing mode. The output function is constructed by multivariate polynomials equations to confuse the plaintext sufficiently. The output Hash size can be adjusted according to different requirements. Storage analysis，forge attack analysis，differential attack analysis and statistic analysis are carried on algorithm. Theoretical analysis and experimental results show that the parallel structure of the algorithm compensates the inefficiency of traditional multivariate polynomial cryptosystems，and it can resist forge attack，differential attack and statistic attack.

【Key words】Hash function；MQ problem；chaotic mapping；cross processing；parallel mode

DOI：10.3969/j.issn.1000-3428.2015.01.024

1　概述

Hash 函数，也称散列函数、凑杂函数，用于将任意长度的消息压缩为固定长度的消息摘要。Hash 函数广泛应用于数据完整性检测、身份认证、数字签名等领域。传统的 Hash 函数采用 Merkle-Damgård 结构，如经典的 MD5[1] 和 SHA-1[2] 算法。然而这类 Hash 函数的安全性能在文献[3-4]中被证实不可靠，且串行处理的结构导致效率不高，研究人员不再满足于传统 Hash 函数。

混沌是一种复杂的非线性动力学系统，它具有初值敏感性、随机性、长期不可预测性等性质，与密码学上扩散与混乱的要求一致，因此诞生了混沌密码学。近年来为了提高 Hash 算法的运行效率，新的混沌 Hash 算法越来越倾向于采用并行处理的模式。文献[5]提出了一种时空混沌的可并行处理 Hash 函数，文献[6]也成功地找到了该算法的碰撞。文献[7]利用分段线性函数和 4 维猫映射构造了一种可并行处理 Hash 函数，但文献[8]发现了该算法难以抵抗伪造攻击。尽管并行处理结构效率很高，但这种并行处理的结构导致明文的各分组之间联系不强，很容易遭受攻击。

基金项目：国家自然科学基金资助项目(61001004)；湖南省教育厅基金资助项目(11B002)；湖南省海外名师基金资助项目(2013008)；湖南省研究生科研创新基金资助项目(CX2013B376)。

作者简介：张文婷(1989－)，女，硕士研究生，主研方向：单向 Hash 函数；龙　敏，教授、博士。

收稿日期：2014-03-03　　**修回日期：**2014-04-03　**E-mail：**yesterdayjuly@qq.com

有限域上 MQ 问题的求解是 NP 难解性问题,利用 MQ 问题构造密码可以抵抗量子计算机的威胁。2007 年 Billet 等人首次将 MQ 问题应用到 Hash 函数中,提出了 MQ-Hash[9]。然而这种基于多变量多项式方程组求解困难性的 Hash 函数存储空间巨大,运行速度缓慢,实用价值不高。而文献[10-11]也指出了直接利用 MQ 问题构造的多变量 Hash 函数的不足。

本文基于混沌映射和多变量多项式方程组构造了一种交叉结构的 Hash 算法,该算法利用并行 Hash 算法和多变量 Hash 算法的优点,抑制了两者分别容易遭到伪造攻击和差分攻击的缺陷。

2　预备知识

2.1　混沌映射

混沌是一种类似无规则、随机的运动,它对初值以及参数极其敏感,在长期内不可预测。这些性质都满足密码学中混乱与扩散的要求相符。本文采用 Henon 映射和分段线性映射来构造 Hash 算法。

(1)Henon 映射

$$\begin{cases} x_{i+1} = y_i + 1 - \alpha x^2 \\ y_{i+1} = \beta x_i \end{cases}$$

其中,当 $\alpha = 1.4, \beta = 0.3$ 时为典型 Henon 映射,典型 Henon 映射是混沌映射。Henon 映射是二维混沌映射,它由 x_i, y_i 共同确定迭代方程,本文算法选择 Henon 映射生成密钥流,加大了算法的密钥空间。

(2)分段线性映射 PLCM

$$x_{i+1} = \begin{cases} x_i / Q & 0 \leqslant x_i \leqslant Q \\ (x_i - Q)/(0.5 - Q) & Q \leqslant x_i \leqslant 0.5 \\ (1 - x_i - Q)/(0.5 - Q) & 0.5 \leqslant x_i \leqslant 1 - Q \\ (1 - x_i)/Q & 1 - Q \leqslant x_i \leqslant 1 \end{cases}$$

当 $x \in [0,1], Q \in (0,0.5)$ 时,系统处于混沌状态。分段线性映射的分布均匀,且输出轨道的自相关函数是 δ 形的。同时它的形式简单,只有一个控制参数,易于实现且速度上有优势。本算法选择 PLCM 映射作为压缩函数。

2.2　MQ 问题

给定一个有限域 $F = GF(q)$ 上 n 个变量 m 个方程二次多项式方程组可以表示为:

$$f_i = \sum_{1 \leqslant j \leqslant k \leqslant n} a_{i,j,k} x_j x_k + \sum_{1 \leqslant j \leqslant n} b_{i,j} x_j + c_i$$

其中,$1 \leqslant i \leqslant m, a_{i,j,k}, b_{i,j}, c_i \in GF(q)$。则 MQ 问题为如下所述,给定方程组 $f = (f_1, f_2, \cdots, f_m)$,每个分量 f_i 都是有限域 $GF(q)$ 上随机选择的 n 个变量的二次方程。对于已知的 $y \in F^m$,求解一个 $x \in F^n$,满足 $y = (f_1(x), f_2(x), \cdots, f_m(x))$,叫做 MQ 问题。

MQ 问题是 NP 难解性问题[12]。本文算法用二次多变量多项式结构作为输出函数,将前面并行的压缩函数隐藏起来,从而达到抵御伪造攻击的目的。

3　Hash 函数的构造

3.1　消息填充

设明文的长度为 m,Hash 值的长度为 n,明文被填充为 512 bit 的整数倍:在明文相应的二进制形式之后补充一个 1,再补充若干个 0。这段 $(100\cdots0)_2$ 串的长度为 l,满足 $(m + l) \bmod 512 = 512 - 16 - 64 = 432$。然后在右边添加 16 bit 的 Hash 值长度 n,最后再添加 64 bit 的消息明文长度 m,如果 $m > 2^{64}$,则取 $m \bmod 2^{64}$。

消息填充结构如图 1 所示。

明文	$(100\cdots0)_2$	Hash值长度	明文长度

图 1　消息填充结构

3.2　算法描述

算法运算在由不可约多项式 $y^8 + y^4 + y^3 + 1$ 定义的 GF(2^8) 上,Hash 值长度可取 192 bit,224 bit,256 bit,384 bit 等。为描述方便,下文输出 Hash 值长度默认为 192 bit。

(1)填充后的明文可分成 s 个 512 bit 的消息块 $M_1, M_2, \cdots, M_s$。对于每个消息块 $M_i (i = 1,2,\cdots,s)$ 再分为 64 个子块 $M_i = block_{1,i}, block_{2,i}, \cdots, block_{64,i}$,其中每个 $block_{j,i}$ 为 8 bit。对于所有 $block_{j,i} (j = 1,2,\cdots,64)$ 经下式线性变换后映射得到 PLCM 映射的参数 $Q_{j,i}$。

$$Q_{j,i} = \frac{1}{6}\left(\frac{block_{j,i}}{256} + \frac{i}{s+1} + \frac{j}{65}\right)$$

(2)算法的密钥为 Henon 映射的一对初值 (xx, yy)。迭代 Henon 映射 50 次后选取后 32 对迭代值 $(xx_\eta, yy_\eta) (\eta = 19,20,21,\cdots,50)$ 进行下式变换得到 PLCM 映射的初值。

$$key_{2(\eta-18)-1} = |1 - |xx_\eta||$$
$$key_{2(\eta-18)} = |1 - |yy_\eta||$$

这 64 个 PLCM 映射的初值分别为 $key_1, key_2, \cdots, key_{64}$,而系统参数是变化的,每轮系统参数取相应的 $Q_{j,i}, (j = 1,2,\cdots,64; i = 1,2,\cdots,s)$,共迭代 $2s$ 次。因这 64 个 PLCM 是并行处理的,所以每一路的操作一样,以 $PLCM_1$ 为例:$PLCM_1$ 前 s 次迭代,参数依次为 $Q_{1,1}, Q_{1,2}, \cdots, Q_{1,s}$;$PLCM_1$ 后 s 次迭代,参数依次为 $Q_{1,s}, Q_{2,s-1}, \cdots, Q_{1,1}$。这 64 个 PLCM 构成交叉结构,即每一轮 $PLCM_j$ 生成的迭代值作为下一轮 $PLCM_{j+1}$ 的初值,而 $PLCM_{64}$ 的迭代值则作为下一轮 $PLCM_1$ 的初值。最后这 64 个 PLCM 同时得到64 个终值,作为二次多变量多项式结构的输入。$2s$ 轮迭代之间的交叉结构如图 2 所示。

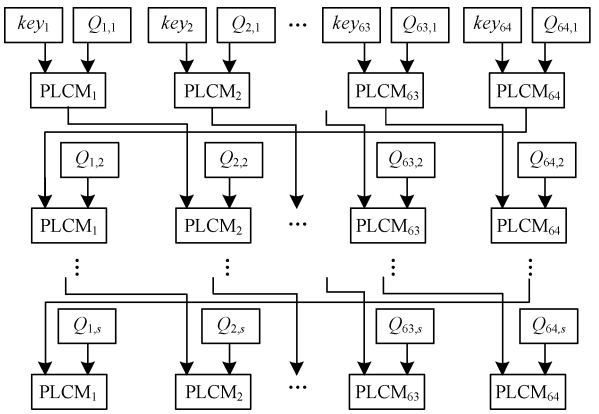


图 2 压缩函数的交叉结构

(3)64 个 PLCM 映射的终值作为二次多变量多项式方程组的 64 个变量 $x_1, x_2, \cdots, x_{64}$。此时 $GF(2^8)$ 上 64 个变量 24 个方程的二次多变量多项式结构可表示为:

$f_k: F^{64} \rightarrow F^{24}$

每个 $f_k(k=1,2,\cdots,24)$ 都为如下形式:

$$f_k = \sum_{1 \leqslant p \leqslant q \leqslant 64} a_{k,p,q} x_p x_q + \sum_{1 \leqslant p \leqslant 64} b_{k,p} x_p + c_k$$

其中,$a_{k,p,q}, b_{k,p}, c_k$ 都是从 $GF(2^8)$ 上随机选择的元素。

这个二次多变量多项式结构将 64 个变量压缩为 24 个变量,而这 24 个变量级联起来得到最终 192 bit的 Hash 值。算法的整体结构如图 3 所示。

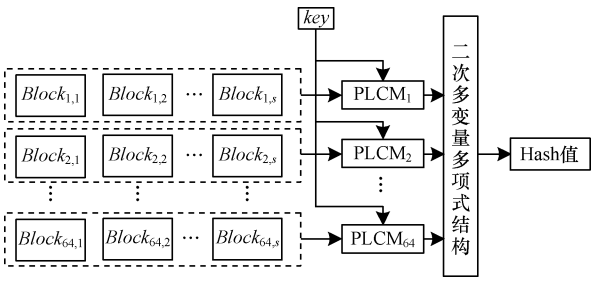


图 3 算法整体结构

4 性能分析与仿真结果

算法的压缩函数可定义为 $C = F(P(key, M))$,其中,F 为多变量多项式方程组;P 为 64 路混沌映射;key 为密钥;M 为消息明文。设每轮函数 P_i 的结果为 $\rho_i = (\rho_{1i}, \rho_{2i}, \cdots, \rho_{64i})$,要找到一对 $M \neq M'$,使得 $C(M) = C(M')$,做如下证明:设第 i 轮时 $\rho_i = \rho_i'$,$i-1$是满足 $\rho_{i-1} \neq \rho_{i-1}'$最大的情况,对于任意的 ρ_{i-1}' 计算得到 $\rho_i' = P_i(\rho_{i-1}', Q_i')$,可推出 $Q_i \neq Q_i'$。假设第 i 轮后消息分组的值完全相同,则此后的迭代轨迹完全一致,在反向迭代至 $2s-i$ 次时,再次使用 Q_i'作为参数计算得到 $\rho_{2s-i+1}' = P_{2s-i}(\rho_{2s-i}', Q_i')$,由于 $Q_i \neq Q_i'$,$\rho_{2s-i} = \rho_{2s-i}'$而可推出 $\rho_{2s-i+1} \neq \rho_{2s-i+1}'$。那么对于往后 $2s-i+1$ 至 $2s$ 轮迭代,根据混沌的性质,其轨迹仍然将会截然不同,从而导致 F 的计算结果不同,即$C(M) \neq C(M')$,因此可以认为算法是无碰撞的。混沌的运动十分复杂,对初值极其敏感,对明文任意微小的改变都会导致迭代值极大的变化,即使在攻击者已知 P_{2s}的最坏情况下,求解 F 也是不可行的,因为 F 是个 NP 难解性问题,从而保证了算法的单向性。

4.1 存储空间

整个算法由混沌迭代和多变量多项式计算 2 个部分组成。第一部分所需存储空间主要是 PLCM 映射的状态值和系统参数,而迭代 64 个 PLCM 映射 $2s$ 次需 $64 \times 2s \times (2 \times 8)$ bit。第二部分为有限域 $GF(2^8)$上的二次多变量多项式方程组,其所需存储空间主要是有限域 $GF(2^8)$的乘法表和随机系数。$GF(2^8)$有 256 个元素,每个元素 8 bits,则乘法表需要 $256 \times 256 \times 8$ bit。多变量多项式方程组有 $nm(n+l)/2$ 个二次项,nm 个一次项和 m 个常数项。表 1 列出不同 Hash 值长度情况下 MQ 部分所占存储空间。

表 1 二次多变量多项式结构所占存储空间

Hash 值长度/bit	随机数	存储空间/KB
192	51 480	114.27
224	60 060	122.65
256	68 640	131.03
384	102 960	164.55

混沌部分所占空间由明文大小决定,多变量多项式部分的存储空间则是固定的。因迭代 Henon 映射 50 次产生的密钥流所占空间极小,此处计算忽略。因此,算法的存储空间主要由明文大小和多变量多项式的随机数决定。一段大小为 $512s$ bit(s 为明文分组数)的明文在 Hash 值长度为 192 bit 的情况下,多变量多项式结构占 114.27 KB 混沌部分占 $0.25s$ KB,共计 $114.27 + 0.25s$ KB。显然,与占用将近 3.5 MB 的 MQ-Hash[9],本算法在存储空间上具有明显优势。这是因为 MQ-Hash 的采用的传统的串行处理结构,每个压缩函数中都要对当前明文分组进行一次多变量多项式计算,而本文的压缩函数为并行的混沌映射而多变量多项式方程组为输出函数,对于任意长度的明文仅作一次多变量多项式计算。

4.2 抗伪造攻击性

伪造攻击是指给定一段消息,攻击者不知其密钥的情况下根据该消息的 Hash 值来构造明文,从而达到攻击的目的。文献[6,8]中指出并行处理的结构导致明文的各分组之间联系不紧密,因而无法抵御伪造攻击。因此,本算法在并行运算前,对明文的每个分组做了如下线性变化来加强各分组的联系:

$$Q_{j,i}=\frac{1}{6}\left(\frac{block_{j,i}}{256}+\frac{i}{s+1}+\frac{j}{65}\right)$$

该变换由 3 个位置因素决定:当前消息块的横向坐标 i、当前消息块的纵向坐标 j 和消息块的总数 s,增减修改明文的任何数据都会不同程度地改变 i,j,s,从而改变了 $Q_{i,j}$ 的值。同时算法又引进了交叉处理,当前混沌映射处理后得到的迭代值在下一轮中并不由自身处理,而是送给相邻的混沌映射处理,这样大大加强了明文不同分组之间的联系,中间值微小的变化会在最终的 Hash 中放大,使得明文充分混淆。此外算法在并行的压缩函数之后,用二次多变量多项式结构作为输出函数。对于文献[6,8]中提出的将上下两行 Hash 值交换后进行异或运算而产生相同结果的攻击方法,由于本算法中多变量多项式结构进一步对并行处理结构的结果充分置乱,隐藏起压并行处理结构的内部状态,从而不再可行。

4.3　抗差分攻击性

MQ 问题是 NP 难解性问题。但是解决 MQ 问题的复杂度取决于方程数 m,变量数 n 和有限域的阶 q。现已存在有效地解决超定多变量多项式方程($n<m$)的方法和置换多变量多项式方程($n=m$)的方法,因此在利用多变量多项式构造密码时通常使用不定方程($n>m$)。本算法在选择 Hash 值长度为 192 bit,224 bit,256 bit,384 bit 时,多变量多项式结构均可表示为 $F^{64}\rightarrow F^{24}$,$F^{64}\rightarrow F^{28}$,$F^{64}\rightarrow F^{32}$ 和 $F^{64}\rightarrow F^{48}$ 的不定方程形式。相比求解复杂度为 $O(q^m)$ 的穷举攻击,文献[13]中提出一种复杂度大大降低的解决不定方程系统的算法:设 $k=\min\left(\frac{m}{2},\left|\sqrt{\frac{n}{2}}-\sqrt{\frac{n}{2}}\right|\right)$,当 k 满足 $2k^2>m-2k$ 时,该算法的复杂度为 $O(q^{m-k})$。可见利用该算法有效求解二次多变量不定方程,对不定方程的取值要严格的限定,该算法仅适用于 q,n,m 很小的情况。按照文献[13]的求解方法在不同的 Hash 值长度情况下解决本算法 MQ 问题部分的复杂度在表 2 列出,可见本文算法的二次多变量多项式结构理论上不可解。

表 2　二次多变量多项式结构的求解复杂度

Hash 值长度/bit	复杂度
192	2^{152}
224	2^{184}
256	2^{216}
384	2^{344}

设 Q 为有限域 F 上 m 个方程 n 个变量的二次方程组为 $f_1,f_2,\cdots,f_m$。对于任意给定的 $\delta=(\delta_1,\delta_2,\cdots,\delta_n)$ 可在多项式时间 $O(mn^2)$ 内找到一对输入 $x=(x_1,x_2,\cdots,x_n)$,$y=(y_1,y_2,\cdots,y_n)$ 发生碰撞,其中,$(x,y)=(x,x+\delta)$[9]。单独使用二次多变量多项式来构造密码是不安全的,无法抵御差分攻击。

差分攻击的是针对分组通过分析明文对的差值对密文对的影响来恢复部分密钥,而本文算法中明文的各分组在经多变量多项式结构处理前先经过交叉的 64 组混沌映射处理,经多轮迭代将初始值扩散到整个相空间,这种对明文充分的置乱很难从最终的迭代值中推算原始消息。而要在本文算法的二次多变量多项式结构部分发现碰撞,必须要求前面的压缩函数部分可逆,而压缩函数采用的混沌映射的性质确保了在输入差分经过数次迭代已均匀分布到所有比特位,因此,在二次多变量多项式结构寻找碰撞是不可行的。

4.4　统计分析

4.4.1　扩散与混乱性质测试

Shannon 引入了扩散与混乱 2 个概念来评估密码系统的性能。对于用二进制表示的 Hash 值,每 1 bit只有 0 或者 1 两种可能,因此理想的 Hash 值扩散效果应该是明文中 1 bit 变化会引起 Hash 值 50% 的变化。

平均变化比特数 $\overline{B}$、平均变化概率 P、变化比特数 B 的均方差 ΔB、P 的均方差 ΔP 4 个统计量如下定义:

$$\overline{B}=\frac{1}{N}\sum_{i=1}^{N}B_i,P=\frac{\overline{B}}{h}$$

$$\Delta B=\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(B_i-\overline{B})^2}$$

$$\Delta P=\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{B_i}{h}-P\right)^2}$$

其中,N 为统计次数;B_i 为第 i 次测试变化的位数;h 为 Hash 值的长度。

取密钥 $xx=1.17$,$yy=-0.28$。对初始文本“Hash function is an algorithm converts variable length message to a fixed length of digest,which is called hash value. Hash function has a wide range of application in information security,such as data integrity authentication, identity authentication and digital signature.”进行 1 000 次测试,每次随机改变文本中 1 bit 的值,并比较改变后的 Hash 值和原始文本的 Hash 值。不同 Hash 值长度下 1 000 次测试相应的平均变化比特数 $\overline{B}$、平均变化概率 P、均方差 ΔB、均方差 ΔP 在表 3 中列出。

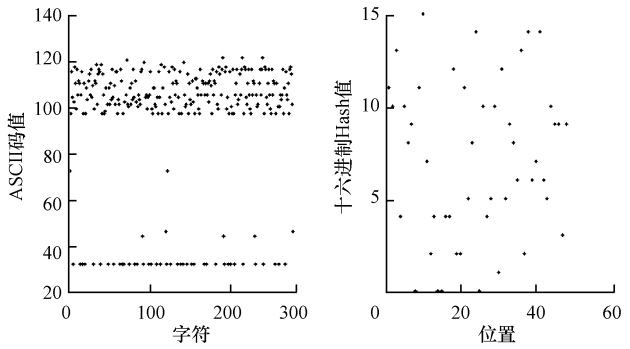
表 3 雪崩测试统计结果

Hash 值长度/bit	$\overline{B}$	P/%	ΔB	ΔP/%
192	94.65	49.22	9.08	4.73
224	112.15	50.07	7.28	3.25
256	127.82	48.19	9.83	4.23
384	190.05	49.49	9.71	2.53

可以看出本算法的$\overline{B}$接近 Hash 值长度的一半，P 也接近的 50% 。而 ΔP 和 ΔB 的值很小，所有统计值都接近理想要求，证明算法的扩散与混乱性能很好。

4.4.2 Hash 值分布

安全的 Hash 算法的一个重要要求是 Hash 值均匀分布。本文对以下消息进行仿真实验："Hash function is an algorithm converts variable length message to a fixed length of digest，which is called hash value. Hash function has a wide range of application in information security，such as data integrity authentication，identity authentication and digital signature." 密钥取 $xx = 1.17$，$yy = -0.28$。通过仿真结果可以看到明文与 Hash 值的分布：图 4(a)为明文的 ASCII 码值分布图，ASCII 码值集中地分布在 90～120 这个范围内；而图 4(b)为十六进制 Hash 值的分布图，Hash 值分布十分分散。由此可知，本文算法得到的 Hash 值有良好的扩散与混乱性能。



(a)明文的ASCII码值分布 (b)十六进制的Hash值分布

图 4 明文和 Hash 值的分布

4.4.3 敏感性分析

为测试本文算法对于明文和密钥的敏感程度，在 7 种不同条件下对 4.5.1 中的文本进行仿真实验。

文本 1：原文。

文本 2：将原文首字母 H 改为 G。

文本 3：将原文中 as 一词改为 is。

文本 4：将原文末尾处句点改为逗号。

文本 5：在原文末尾添加一位空格。

文本 6：将密钥 yy 取值由 -0.28 改为 $-0.280\ 000\ 000\ 000\ 000\ 1$。

文本 7：将密钥 xx 取值由 1.17 改为 $1.170\ 000\ 000\ 000\ 000\ 1$。

仿真所得 192 bit 的 Hash 值长度的十六进制形式如下：

文本 1：BAD4A890BF7240044C22B58E0A45A1C5986D2E67E65A9939

文本 2：27F08D4ECBC371221D35193A4D797391E4CE56077B0A26D5(改变 94 位)

文本 3：0BC2E9D74577C01E611F6480C51C63F43ECCC5DD136CFB7F(改变 91 位)

文本 4：0F034E33295F561EF088BFF984B55ED96989CC988750FE02(改变 106 位)

文本 5：CEE066D48D20D8F2DA3C1924A69539735C39FC711CA64547(改变 96 位)

文本 6：400543850934F5EB02D3797DB6BB53CBE2F06E62CF721A02(改变 104 位)

文本 7：F093000A8363508DD75F621B782CEA4A6C6D6397C30A546D(改变 88 位)

图 5 所示为相应的 Hash 值二进制形式的图形分布，可以看出对明文和密钥的任何细微改变都会对 Hash 值产生巨大的影响，证明算法对明文和密钥的敏感程度良好。

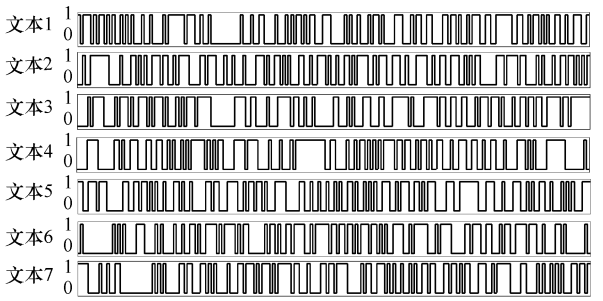


图 5 7 种条件下二进制 Hash 值比较

5 结束语

本文结合混沌理论的复杂性和 MQ 问题的难解性，提出一种新的交叉迭代的 Hash 算法，算法占用空间小、处理速度快，另外，可根据不同需求调节 Hash 值长度。仿真实验和理论分析证明本文算法满足 Hash 算法的安全要求，且具有处理大容量数据的潜力。今后的工作重点是以该算法为基础研究在云端进行消息认证的方法。

参考文献

[1] Rivest L. The MD5 Message digest Algorithm [S]. RFC 1321，1992.

[2] NIST. FIPS PUB 180-2-1993 Secure Hash Standard[Z]. 1993.

[3] Wang Xiaoyun，Yu Hongbo. How to Break MD5 and Other Hash Functions [C]//Proceedings of EUROCRYPT'05. Berlin，Germany：Springer-Verlag，2005：19-35.

(下转第 149 页)