# Analysis of a Novel Hash Function Based upon Chaotic Nonlinear Map with Variable Parameter

Mahdi Nouri, Sajjad Abazari Aghdam, Payam Pourmahdi, and Mahyar Safarinia

***Abstract*—In this paper a parallel Hash algorithm construction based on chaotic maps with variable parameters is proposed and analyzed. The two core characteristics of the recommended algorithm are parallel processing mode and chaotic behaviors. A chaos system also creates a random behavior subsequently, but at the same time a chaos system is wholly deterministic. Moreover in this paper, an algorithm for one way hash function construction based on chaos theory is introduced. The proposed algorithm contains variable parameters dynamically obtained from the position index of the corresponding message blocks. Theoretical analysis and computer simulation indicate that the algorithm can assure all performance requirements of hash function in an efficient and flexible style and secure against birthday attacks or meet-in-the-middle attacks, which is good choice for data integrity or authentication.**

***Index Terms*—Hash function; Two-dimensional coupled map lattices; Spatiotemporal chaos; Chaotic nonlinear map; variable parameter**

## I. INTRODUCTION

AS one of the cores of cryptography, Hashing is a basic technique for information security. After the conventional Hash function such as MD5, SHA was successfully attacked, the research on the design of secure and efficient Hash function stays a research hotspot. The ideal hash function must have the following main properties:

- Simplicity of computation: It should be simple to calculate a hash for any given data.
- Pre-image resistance: It should be enormously difficult or almost impossible in a practical sense to calculate a text that has a given hash i.e. given hash value H, it should be very difficult to find any m such that a H = hash (m).
- Collision-resistance: It should be implausible that two different messages $m_j$ and $m_2$, however close, have hash ($m_1$) = hash ($m_2$).
- Fixed size output: The outcome message digest/cipher should be of fixed size. A hash of a short message will create the same hash size digest as a hash of a full set of encyclopedias.
- Security: The hash algorithm itself does not need to be kept secret. It is made available to the public. Its security comes from its ability to create one-way hashes.

Because of being some shortcomings in conventional Hash function constructions, the newly proposed chaos-based Hash functions reveal an attractive design direction [1– 11]. In 2008, Wang et al. [6] proposed a one-way Hash function construction based on two dimensional coupled map lattices, which utilized a two dimensional coupled map lattices with features leading to the largest Lyapunov exponent. In the same year, Yang et al. [7] proposed a one-way Hash function construction based on chaotic map network. A novel combined cryptographic and Hash algorithm based on chaotic control character was utilized [8] in 2009. However, all of the Hash algorithms declared above cannot be processed in a parallel mode, which can wholly progress the efficiency and speed of the Hash function.

Recently, spatiotemporal chaos has been magnetizing more and more interests among researchers in the fields of mathematics, physics, and engineering. Compared with simple chaotic maps, spatiotemporal chaos has two supplementary merits for cryptographic purpose. Due to the finite computing precision, chaotic orbits will ultimately become periodic. The period of spatiotemporal chaos is longer than that of simple chaotic maps [6, 12]. In particular, the period of chaotic orbits created by a system with a great number of chaotic coupled oscillators is too long to be reached in practical communications. Consequently, periodicity can be practically avoided in spatiotemporal chaotic systems [6, 13]. Furthermore, spatiotemporal chaotic system is a high dimensional chaotic system, which has a number of positive Lyapunov exponents that guarantee multifarious dynamical behavior. It is more difficult or even impossible to forecast the time series made by spatiotemporal chaos.

Coupled map lattice model presents spatiotemporal nonlinear system a qualitative description through making space and time discrete and keeping states variable

M. Nouri is with the Dept. of Electrical Engineering, Iran University of Science & Technology, Iran (e-mail: mnuri@elec.iust.ac.ir)

S. Abazari Aghdam is with the Dept. of Electrical Engineering, South Tehran Islamic Azad University, Iran (e-mail: sajjadabazarei @gmail.com)

P. Pourmahdi is with the Electrical Engineering Department, A.B.A Institude of Higher Education, Tehran, Iran (e-mail: payamict67m@gmail.com).

M. Safarinia is with the Electrical Engineering Department, A.B.A Institude of Higher Education, Tehran, Iran (e-mail: mahyar1986@gmail.com).



http://sites.google.com/site/ijcsis/
ISSN 1947-5500

continuous.

There are some great benefits of spatiotemporal hyper chaos in comparison with short dimensional chaos. In the former

case there exist a huge number of spatial sites. When chaos is applied for treating message, each of which takes chaotic action and serves as a message operation, and then the message can be performed simultaneously and in parallel by many subunits, and thus the efficiency of message healing can be greatly increased.

The enlarged message blocks are converted into the resultant ASCII code values as the iteration times by the proposed algorithm, which the algorithm iterates the chaotic nonlinear map, continuously, with the variable parameters dynamically attained from the position index of the corresponding message blocks to create decimal portions and after that rounds the decimal portions to integers, and finally flows these integers to construct intermediate Hash value. Final Hash value with the length of 128-bit is created by four outputs of last iteration.

## II. Analysis of the Chaotic Nonlinear Map

The two-dimensional chaotic nonlinear map in the algorithm is represented as (1):

$$f(x_{i+1}, y_{i+1}) = \begin{cases} (x_i + e^{k \sin\left(\frac{x_i}{k}\right)}) - \left[x_i + e^{k \sin\left(\frac{x_i}{k}\right)}\right] \\ (y_i + x_{i+1}) - [y_i + x_{i+1}] \end{cases} (1)$$

Where $x_i, y_i \in [0,1)$ and $k$ is the controlled variable and belongs to (0,10). The map is nonlinear and the parameter $k$ ensures that the map runs in a chaotic status when $0<k<10$. It transforms an interval (0,10) onto itself and includes only one parameter $k$.

The chaotic nonlinear map also has the same belongings to proposed map that are fit for composing Hash function. The form of the map is complicated and the equation involved is nonlinear. Figure 1 shows the simulation of the chaotic nonlinear map iterating 64 times with the initial values $x_0, y_0 = 0.3423$ and parameter $k = 0.4$.

The map has some properties, which are appropriate for constructing the Hash function, such as initial value sensitivity and parameter sensitivity, with variable parameter $k$ valued in the interval (0,1). Figure 1 displays the chaotic iteration property with variable parameter a valued in the interval (0,10), which initial values are $x_0, y_0 = 0.3423$.

The map has some properties, which are appropriate for constructing the Hash function, such as initial value sensitivity and parameter sensitivity, with variable parameter $k$ valued in the interval (0,1). Figure 2 displays the chaotic iteration property with variable parameter a valued in the interval (0, 10), which initial values are $x_0, y_0 = 0.3423$.

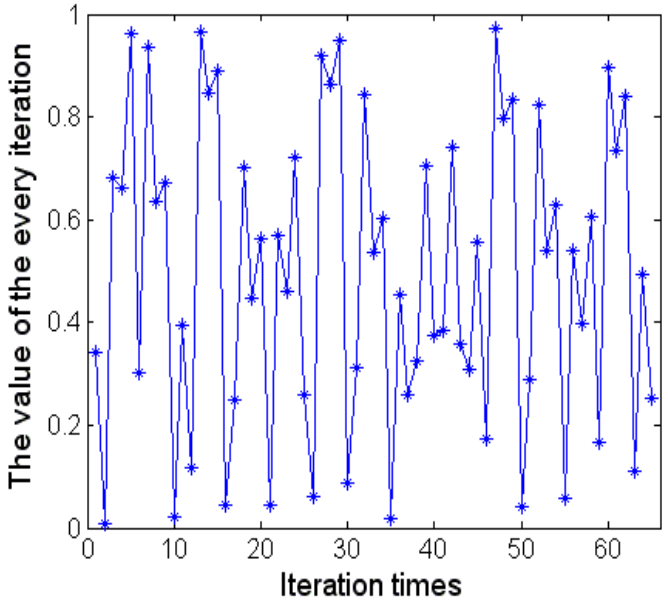



Fig. 1 Iteration property with changeable parameter a when $x_0, y_0 = 0.3423$

Only after several iterations, the sensitivity of chaotic systems can be shown. Take a test data as an example to test the sensitivity, the difference between the two initial values of each group, which is denoted by e, meets the condition $|e|<=10^{-12}$. Choose 8, 16, 32 and 64 respectively as the number of Iterations. The difference of the final value can be shown in Table 1. In this experiment, iteration operation on 2000 homogeneously distributed interval data in [0, 1) is performed several times, also choose 8, 16, 32 and 64 respectively as the number of iterations, list the statistics of the final value in each region.

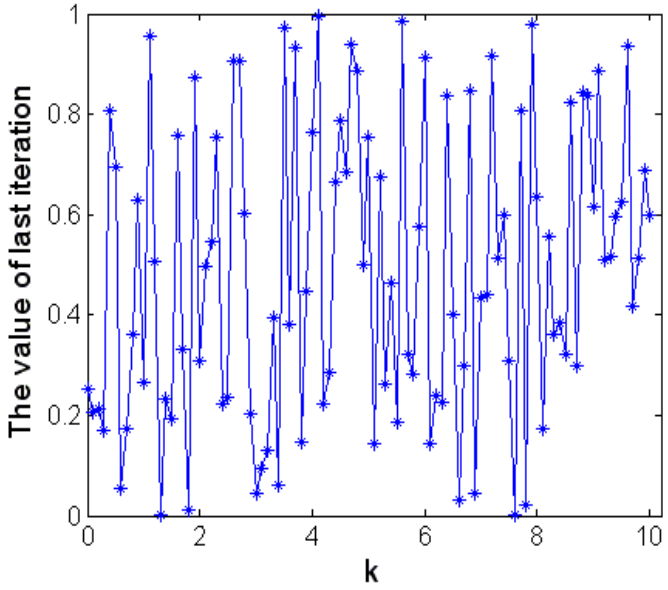



Fig. 2 Iteration property with changeable parameter k when $x_0, y_0 = 0.3423$

TABLE I



http://sites.google.com/site/ijcsis/
ISSN 1947-5500

THE DATA OF SENSITIVITY TEST FOR N=8, 16

| *N* | *8* | *16* |
|---|---|---|
| **0.10000000000** | 0.953585271946239 | 0.151651625422085 |
| **0.10000000001** | 0.953585275641486 | 0.151717969326340 |
| **0.30000000000** | 0.690422342779915 | 0.521789230012578 |
| **0.30000000001** | 0.690422359899307 | 0.521834907935783 |
| **0.50000000000** | 0.312717379924095 | 0.302664392758495 |
| **0.50000000001** | 0.312717425838585 | 0.302797948128847 |
| **0.70000000000** | 0.547613280987791 | 0.733487200933088 |
| **0.70000000001** | 0.547613307788567 | 0.733537006543614 |
| **0.90000000000** | 0.703912822319106 | 0.293836589632238 |
| **0.90000000001** | 0.703912839754912 | 0.293891061238458 |

TABLE II
THE DATA OF SENSITIVITY TEST FOR N=32, 64

| *N* | *32* | *64* |
|---|---|---|
| **0.10000000000** | 0.807604920204249 | 0.069616762312453 |
| **0.10000000001** | 0.439807354993843 | 0.676225569091725 |
| **0.30000000000** | 0.514317274820386 | 0.245180491047975 |
| **0.30000000001** | 0.858467722762351 | 0.171344293042810 |
| **0.50000000000** | 0.793420964172318 | 0.265449981564812 |
| **0.50000000001** | 0.011091730191061 | 0.703895294281160 |
| **0.70000000000** | 0.528830105657481 | 0.808479816199454 |
| **0.70000000001** | 0.874130283588605 | 0.769349336615293 |
| **0.90000000000** | 0.494016261781210 | 0.996962527902671 |
| **0.90000000001** | 0.031917255574683 | 0.418831604596437 |

As can be seen from Table I, II, the sensitivity to initial values demonstrates obviously after 32 iterations. Therefore, in this algorithm, taking into account both the speed and security, we make the number of iterations range from 61 to 90.

## III. HASH FUNCTION BASED ON STANDARD MAP

The whole structure of the algorithm can be shown in Figure 3.

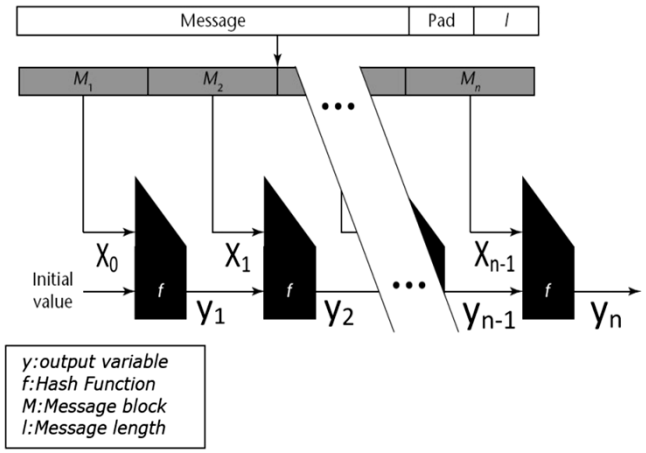



Fig. 3 Whole algorithm structure

### A. Message expansion

Message expansion is significant and necessary; because it greatly improves the sensitivity of each bit in original message to the final Hash value [14]. The plaintext is an arbitrary message that is conveyed in a matrix M, for simple enlightenment of the extended message. Assume that M is a N ×16 plain message matrix, each element with a size of 32 bits.

1) *Padding the message*: The purpose of padding is to ensure the padded message being a multiple of 128 bits. Suppose the total length of message is $W$ bytes, compute $d = (W \bmod 64)$, $0 \le d \le 12$. Pad as follows: if $12 \le d < 16$ then pad $12 - d$ bytes, otherwise pad $28 - d$ bytes, the bytes been padded come from the head of message. The last four bytes are padded with message length. This method ensure at least one byte head of message been padded.

2) *Parsing the padded message*: The padded message is parsed into $N$ 128-bit blocks, $M_1, M_2, \ldots, M_N$. $M_j (1 \le j \le N)$ is parsed into four 32-bit words

$$M_j = [m_{j,1}, m_{j,2}, m_{j,3}, m_{j,4}]$$

3) *Setting the initial values*: The initial values $IV$ is the following eight 32-bit words in hex same as SHA-256 [3]:

$$IV = 2^{-32} \times \begin{cases} 6A09E667 \\ BB67AE85 \\ 3C6EF372 \\ A54FF53A \end{cases} \quad (2)$$

4) *Float point number representation:* The new float point number representation in this paper is to represent each float point number between 0 and 1 with a 32-word $(b_1 b_2 \ldots b_{32})$ as follow:

$$b_1 \times \frac{1}{2} + b_2 \times \left(\frac{1}{2}\right)^2 + \cdots + b_{32} \times \left(\frac{1}{2}\right)^{32} \quad (3)$$

### B. Algorithm for generating the hash

Input of the algorithm can have an input of arbitrary length output of the algorithm has a fixed length of 128 bits. Give a message M with length L.

Take each letter of M as a plaintext block. Transform each plaintext block to the corresponding ASCII numbers; the ASCII numbers create the $x_j, y_j$ which are the inputs of chaotic nonlinear map.

Compression function inputs consider 16 lattice spaces. Let the initial iterative value of these inputs are:


http://sites.google.com/site/ijcsis/
ISSN 1947-5500

$$\begin{cases} x_{j,1} = \left(\frac{m_{j,1}}{10^3}\right) + \left(\frac{m_{j,2}}{10^6}\right) + \left(\frac{m_{j,3}}{10^9}\right) + \left(\frac{m_{j,4}}{10^{12}}\right) \\ x_{j,2} = \left(\frac{m_{j,5}}{10^3}\right) + \left(\frac{m_{j,6}}{10^6}\right) + \left(\frac{m_{j,7}}{10^9}\right) + \left(\frac{m_{j,8}}{10^{12}}\right) \\ x_{j,3} = \left(\frac{m_{j,9}}{10^3}\right) + \left(\frac{m_{j,10}}{10^6}\right) + \left(\frac{m_{j,11}}{10^9}\right) + \left(\frac{m_{j,12}}{10^{12}}\right) \\ x_{j,4} = \left(\frac{m_{j,13}}{10^3}\right) + \left(\frac{m_{j,14}}{10^6}\right) + \left(\frac{m_{j,15}}{10^9}\right) + \left(\frac{m_{j,16}}{10^{12}}\right) \end{cases} \quad (4)$$

From $\{ x_i\}$ and $\{ y_i\}$, 4 groups of $(y_i)$ can be reached. Determine the 32-bits Hash value by the position of the coordinates $(y_i)$ falling into the region of [0, 1), then, finally, juxtaposes these bits from left to right to get a 128-bit hash value.

TABLE III
ALGORITHM FOR GENERATING THE HASH

| Step | Operation |
|---|---|
| 1 | $q \leftarrow 1, j \leftarrow 1, i \leftarrow 2q-1$ go to Step 2 |
| 2 | $k \leftarrow 1/\Pi$<br>$(x_{j,i+4}, y_{j,i+4}) \leftarrow f(x_{j,i}, y_{j,i}), i \leftarrow i+1$ |
| 3 | *if* $i \leq 2k+2$ go to Step 2 |
| 4 | $p \leftarrow 2, k+1 \leftarrow k$, *if* $k \leq ka$ go to Step 2 |
| 5 | $i_1 \leftarrow 23, x'_{j,2ka_{(j)+2}} \leftarrow f$<br>*if* $f_{1,p-1} = 1$<br>$i_p \leftarrow (1.1)^p + [i_{p-1}]$<br>*elseif* $i_p \leftarrow i_{p-1}, p+1 \leftarrow p$<br>*end* |
| 6 | *if* $p < 33$ go to Step 5 |
| 7 | $ka_I$=101<br>$ka_{j+1} \leftarrow ([a_{33}]) mod\ 23 + 61$ |
| 8 | $y_{j+1,1} \leftarrow y_{j,2ka_{j+3}}$<br>$y_{j+1,2} \leftarrow y_{j,2ka_{j+4}}$<br>$y_{j+1,3} \leftarrow y_{j,2ka_{j+5}}$<br>$y_{j+1,4} \leftarrow y_{j,2ka_{j+6}}$ |

## IV. PERFORMANCE ANALYSIS

A. *Hash results of messages:* The uniform distribution of Hash value is one of the most important properties of Hash function, which is directly related to the security of Hash function. Simulation experiment has been done on the following paragraph of message:

**Condition 1:***"A hash function is a fundamental building block of information security and plays an important role in modern cryptography. It takes a message as input and produces an output referred to as a hash value. Generally, hash functions can be classified into two categories: unkeyed hash functions for data integrity, and keyed hash functions, usually known as message authentication code (MAC)."*

**Condition 2:** Change the first character A in the original message to D.

**Condition 3:** Change the word function in the original message to functions.

**Condition 4:** Change the full stop at the end of the original message to a comma.

**Condition 5:** adjoin a blank space to the end of the original message.

The corresponding hash values in hexadecimal format are provided as follows, followed by the equal number of different bits compared with the hash value achieved under Condition1:

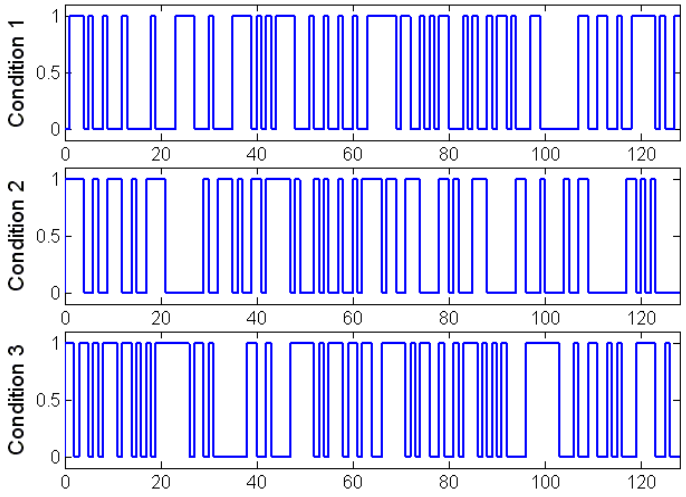
Condition 1: E2114874875F8429FD4D82D260899C79

Condition 2: F4E4E10279D7152DB93C2E0C80910650

Condition 3: B57BAFB50C48FABDC7D4ABA0F746A872

Condition 4: 6AF0343ABCD3BA64671C36E45EE769EA

Condition 5: 53C456373FAF7A3DA2B370915E50170D





http://sites.google.com/site/ijcsis/
ISSN 1947-5500

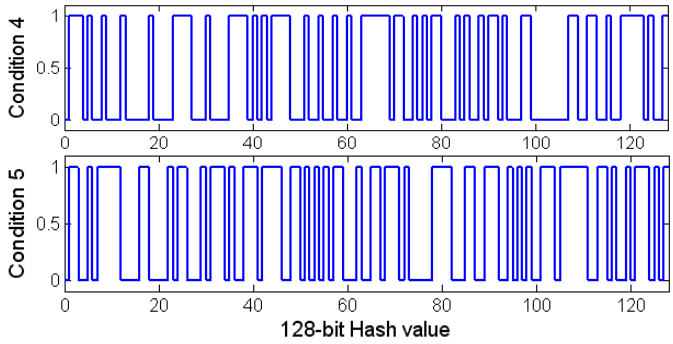



Fig. 4 Hash values under different conditions

The corresponding graphical display of binary sequences is shown in Fig. 4. The simulation results designate that a little difference in the message causes vast changes in the final hash value.

### B. *Statistic analysis of diffusion and confusion*

Confusion and diffusion are two essential design criteria for encryption algorithms, including hash functions. Shannon initiated diffusion and confusion in order to conceal message redundancy [15,16]. Hash function, like encryption system, requires the plaintext to diffuse its effects into the whole Hash space. This means that the correlation between the message and the corresponding Hash value should be as small as possible.

Diffusion means spreading out the influence of a single plaintext symbol over many ciphertext bits so as concealing the statistical structure of the plaintext. Confusion means the utilizing of transformations to make difficult the dependence of ciphertext statistics on plaintext statistics. In the hash value in binary format each bit can be only 0 or 1. Therefore, the perfect diffusion effect should be that any minute change in the initial condition leads to a 50% changes, probability of each bit. Regularly six statistics are defined as follows:

Minimum changed bit number:

$$B_{min} = \min\left(\{B_i\}_1^N\right) \tag{5}$$

Maximum changed bit number:

$$B_{max} = max(\{B_i\}_1^N) \tag{6}$$

Mean changed bit number:

$$\bar{B} = \frac{1}{N}\sum_1^N B_i \tag{7}$$

Mean changed probability:

$$P = \frac{\bar{B}}{128} \times 100 \tag{8}$$

Standard variance of the changed bit number:

$$\Delta B = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(B_i - \bar{B})^2} \tag{9}$$

Standard variance of the changed probability:

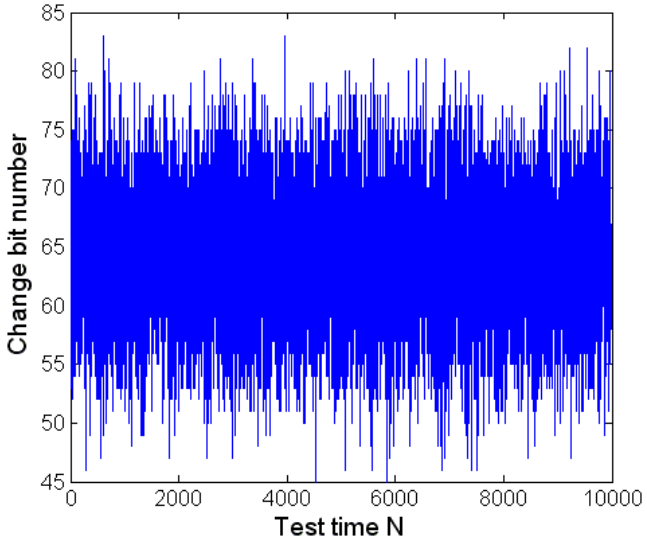
$$\Delta P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\frac{B_i}{128} - P)^2} \times 100 \tag{10}$$

Where $N$ is the total number of tests and $B_i$ is the number of changed bits in the $i_{\text{th}}$ test.

The following diffusion and confusion test has been made that: A paragraph of message is arbitrarily chosen and the corresponding hash value is created. Then a bit in the message is arbitrarily chosen and toggled and a new hash value is created. At last, the two hash values are compared with each other.

This kind of test is performed N times, and the corresponding distribution of changed bit number is plotted in Fig. 5, where $N$ = 10,000. clearly the changed bit number corresponding to 1 bit changed message concentrates around the perfect changed bit number, i.e., 64 bits. It indicates that the algorithm has very strong capability for diffusion and confusion.

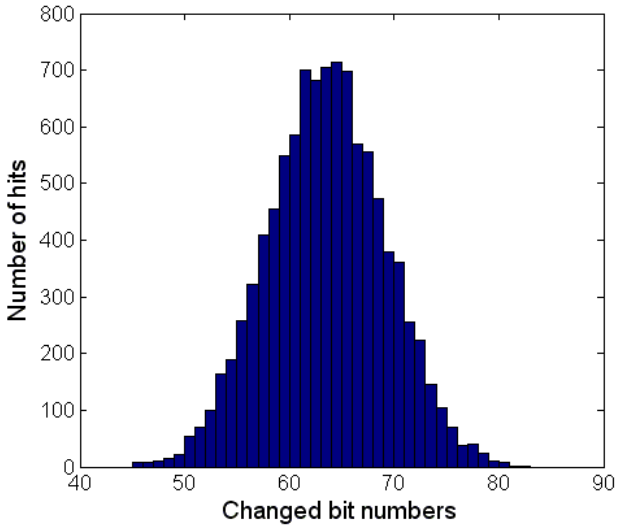
The test results in $N$ = 256, 512, 1024, 2048, 5000, 10,000 are listed in Table IV respectively. The following conclusion was found that, the mean changed bit number $\bar{B}$ and the mean changed probability $P$ are both very close to the ideal value 64 bits and 50%, based on the results. While ΔB and ΔP are extremely small, which indicate the diffusion and confusion capability are very constant. Therefore a small difference in the plaintext will cause great changes in the hash value, which donates to the high plaintext-sensitivity of the proposed hash function. This property is significant in maintaining the secrecy against statistical attacks.




(a)



http://sites.google.com/site/ijcsis/
ISSN 1947-5500




(b)

Fig. 5 Distribution of changed bit number: (a) plot of Bi, (b) histogram of Bi

TABLE IV
STATISTICAL PERFORMANCE OF THE PROPOSED ALGORITHM

| N | 128 | 256 | 512 | 1024 | 2048 | 5000 | 10,000 |
|---|---|---|---|---|---|---|---|
| $B_i$ | 64 | 64.05 | 63.9 | 63.96 | 64.08 | 64.09 | 64.04 |
| ΔB | 5.64 | 5.45 | 5.69 | 5.47 | 5.56 | 5.6 | 5.60 |
| P | 50 | 50.04 | 49.92 | 49.97 | 50.06 | 50.07 | 50.03 |
| ΔP | 4.4 | 4.26 | 4.44 | 4.27 | 4.34 | 4.38 | 4.37 |
| $B_{min}$ | 48 | 46 | 46 | 46 | 45 | 44 | 44 |
| $B_{max}$ | 77 | 77 | 82 | 83 | 83 | 83 | 83 |

### C. *Analysis of speed*

The proposed algorithm is approximately comparative to the length of original message, the iteration steps range from 61 to 90, because the step of each round is arbitrary, the whole iteration steps of this algorithm are estimated approximately. While customary Hash algorithms like MD5, SHA need to structure the primitive plaintext together to the unchanging length. When original message is very small, the existing algorithms still need to do lots of computation, while the proposed algorithm only needs a few steps of iterations to save the computing time.

### D. *Analysis of collision resistance*

The following test is performed to conduct quantitative analysis on collision resistance [6,17,18]: first, the Hash value for a paragraph of message randomly chosen is generated and stored in ASCII format. Then a bit in the paragraph is selected randomly and toggled, and thus a new Hash value is then generated and stored in the same format. Two Hash values are compared, and the number of ASCII characters with the same value at the same location in the Hash values, namely, the number of hits, is counted. A plot of the distribution of the number of hits is given in Fig. 6, and we can see there are 26 tests hit twice and 432 tests hit once, while in 9,542 tests, no hit occurs. It is noticed that the maximum number of equal character is only 2 and the collision is very low.

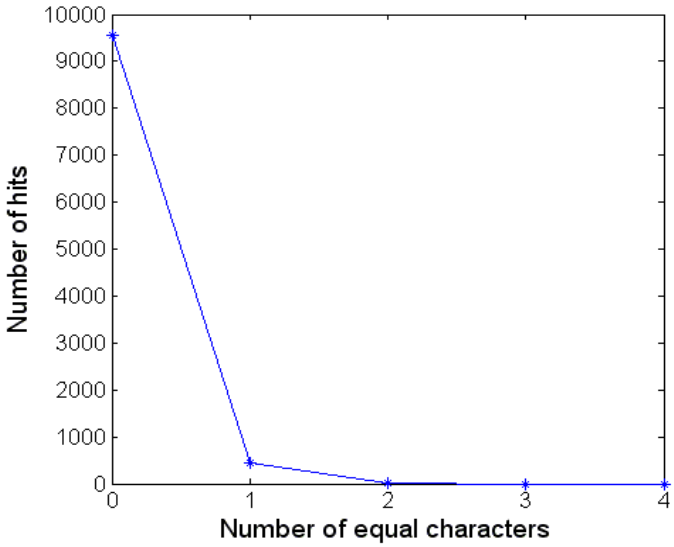



Fig. 6 Distribution of the number of ASCII characters with the same value at the same location in the Hash value

Where $e_i$ and $e'_i$ is the $i_{th}$ ASCII character of the original and the new hash value, respectively. The function $t(\ )$ converts the entries to their equivalent decimal values. This kind of collision test is performed 10,000 times. The maximum, mean, and minimum values of d are listed in Table V. The distribution of the number of ASCII characters with the same value at the same location in the hash value is given in Fig. 9. Notice that the maximum number of equal characters is only 2 and the collision probability is very low.

$$d = \sum_{i=1}^{N} | t(e_i) - t(e'_i) | \qquad (11)$$

TABLE V
ABSOLUTE DIFFERENCE OF TWO HASH VALUES

| *Absolute difference* | *Maximum* | *Minimum* | *Mean* |
|---|---|---|---|
| Xiao's scheme | 2221 | 696 | 1506 |
| Zhang's scheme | 2022 | 565 | 1257 |
| MD5 | 2074 | 590 | 1304 |
| Proposed scheme | 2313 | 697 | 1626 |
| SHA-1 | 2730 | 795 | 1603 |


http://sites.google.com/site/ijcsis/
ISSN 1947-5500

### *E. Analysis of meet-in-the-middle resistance*

Meet-in-the-middle attack [10,11] means to find a contradiction through looking for a suitable substitution of the last plaintext block. For instance, $M = [M_1, M_2, \dots, M_N]$ the expected contradicted one is $M' = [M_1, M_2, \dots, M'_N]$, That is, the attack process is just to replace $M_N$ with $M'_N$ and keep the final Hash value H(M) unchanged. Therefore, it is very difficult to exchange $M_n$ with $M'_n$. If it does as above instance, it will create different Hash value. As a result, the proposed algorithm can resist meet-in-the-middle attack.

### *F. Statistic analysis of diffusion and confusion for variable parameter*

Here into, k is the controlled variable. The certain critical value of k is 0.7.The followed is some conclusion on chaotic nonlinear map. By varying this parameter as can be seen in the Fig. 7, Fig. 8 and Fig. 9, the Statistic analysis of diffusion and confusion are near the same for changing this parameter in 10000 tests and it means this algorithm has a stable manner in all variable parameters.

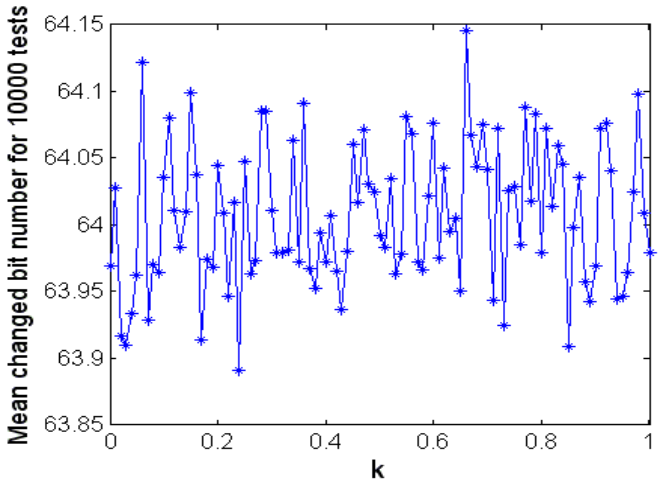



Fig. 7 Mean changed bit number for changing k

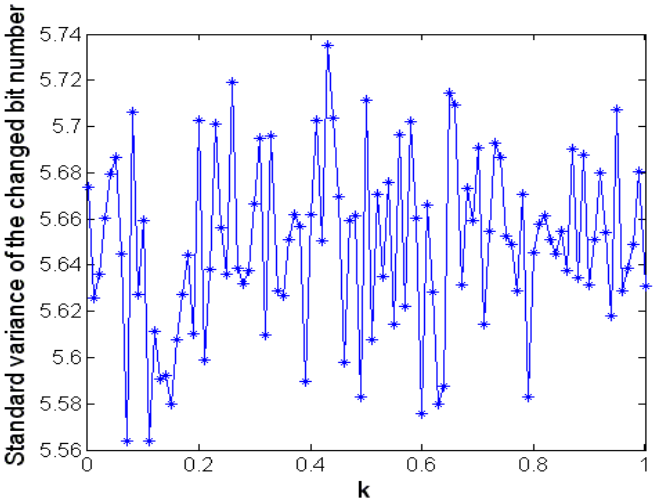



Fig.8 Standard variance of the changed bit number for changing k

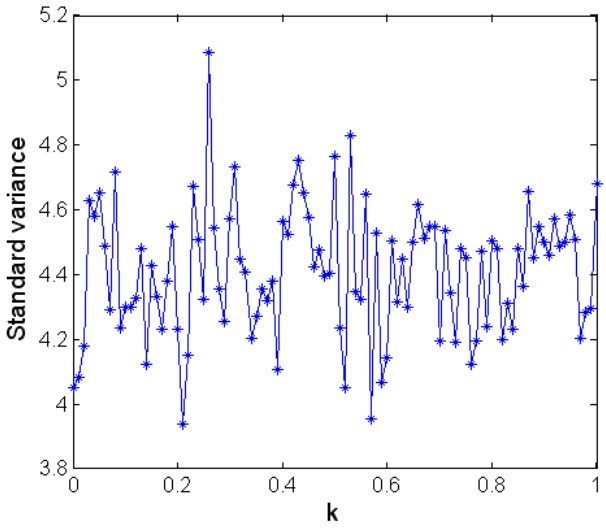



Fig.9 Standard variance of the changed probability for changing k

## V. CONCLUSION

Based on the two-dimensional chaotic nonlinear map with variable parameter, a parallel Hash algorithm structure is proposed and analyzed. The algorithm converts the expanded message blocks into the equivalent ASCII code values. The two initial inputs and steps of iterations are generated by last round of iteration, which iterates the chaotic nonlinear map and wholly increases the rise influence of Hash function, and makes the final Hash value has high sensitivity to the initial values, increase the security of Hash function. The analysis designates that the algorithm can meet all the requisites of the Hash function efficiently. And the algorithm is easy to realize a swift and practical program to Hash function structure. The length of the final Hash value generated by this algorithm is 128 bits. Theoretical analysis and computer simulation signify that the proposed algorithm presents several interesting features, such as high message, good statistical properties, collision resistance, and secure against meet-in-the-middle attacks that can satisfy the performance requirements of Hash function. Furthermore the proposed algorithm can give some extra advantages for having controllable regulator by the variable parameters, where as the results of experiment do not change. It shows that this algorithm has a stable manner in all variable parameters.

## REFERENCES

[1] KW.Wong, " A combined chaotic cryptographic and Hashing scheme" , Phys Lett A 307 , 2003 , pp. 292–298

[2] D .Xiao, XF. Liao and SJ.Deng , "One-way Hash function construction based on the chaotic map with changeable-parameter" , Chaos Solitons Fractals 24 , 2005 , pp.65–71

[3] X .Yi , "Hash function based on chaotic tent maps" , IEEE Trans Circuits Syst II Express Briefs 52 , 2005 , pp.354–357

[4] SG .Lian, JS .Sun and ZQ .Wang , "Secure Hash function based on neural network" , Neurocomputing 69 , 2006 , pp.2346–2350



http://sites.google.com/site/ijcsis/
ISSN 1947-5500

[5] JS .Zhang, XM .Wang and WF. Zhang , "Chaotic keyed Hash function based on feedforward–feedback nonlinear digital filter" , Phys Lett A 362 , 2007 , pp.439-448
[6] Y .Wang, XF .Liao, D. Xiao and KW. Wong , "One-way Hash function construction based on 2D coupled map lattices" , Inform Sci 178 , 2008 , pp.1391–1406
[7] HQ .Yang, KW .Wong, XF .Liao, Y .Wang and DG .Yang , "Oneway Hash function construction based on chaotic map network" , Chaos Solitons Fractals 41 , 2009 , pp.2566–2574
[8] SJ.Deng, YT. Li and D .Xiao , "Analysis and improvement of a chaos-based Hash function construction" , Commun Nonlinear Sci Numer Simulat 15 , 2009 , pp.1338–1347
[9] YT .Li, SJ. Deng and D .Xiao , "A novel Hash algorithm construction based on chaotic neural network" , Neural Comput Appl 20 , 2011 , pp.133–141
[10] YT .Li, D .Xiao and SJ.Deng , "Hash function based on the chaotic look-up table with changeable parameter" , Intern J Modern Phys B (In press) , 2010
[11] D .Xiao, XY .Liao and KW .Wong , "Improving the security of a dynamic look-up table based chaotic cryptosystem" , IEEE Trans Circuits Sys II Express Briefs 53 , 2006 , pp.502–506
[12] S. Wang, W. Liu, H. Lu, et al., "Periodicity of chaotic trajectories in realizations of finite computer precisions and its implication in chaos communications" , International Journal of Modern Physics B 18 , 2005 , pp. 2617–2622.
[13] P. Li, Z. Li, W.A. Halang and G. Chen , "A multiple pseudorandom-bit generator based on a spatiotemporal chaotic map" , Physics Letters A 349 , 2006 , pp.467–473.
[14] CN .Zhang and CR .Lai , "A systematic approach for encryption and authentication with fault tolerance" , Comput Netw 45 , 2004 , pp.143–154
[15] M.K.Viswanath and A.R.Deepti , "A New Approach to a Secure Cryptosystem using the Microcontroller", Journal of Information Assurance and Security, 2006 , pp. 229-236,
[16] Chiang-Lung Liu , "A JPEG-Compliant Method for Image Authentication and Recovery" , Journal of Information Assurance and Security , 2006 , pp. 229-236,
[17] NIST Brief Comments on Recent Cryptanalytic Attacks on Secure Hashing Functions and the Continued Security Provided by SHA-1, 2004. http://csrc.nist.gov/hash_standards_comments.pdf.
[18] Secure Hash Standard, Federal Information Processing Standards Publication (FIPS PUB) 180-2, 2002.