Software Engineering and Applications 软件工程与应用, 2015, 4, 34-43
Published Online June 2015 in Hans. http://www.hanspub.org/journal/sea
http://dx.doi.org/10.12677/sea.2015.43005

Hans汉斯

The Detection of Pseudo-Random Sequence Generated by RC4 Algorithm Based on Frequency Test within a Block

Heyuan Chen[1], Jeffrey Zheng[2]

[1]Department of Information Security, School of Software, Yunnan University, Kunming Yunnan
[2]Key Lab of Yunnan Software Engineering, Kunming Yunnan
Email: heyuanchen91@163.com

Received: May 19th, 2015; accepted: Jun. 6th, 2015; published: Jun. 10th, 2015

Abstract

The security of stream cipher mainly depends on its randomness of pseudo-random sequence. NIST has provided 16 methods for pseudo-random sequence detection, however they are only capable of detecting one selected segment. Hence, the randomness testing method for large numbers of segments needs further systematic research. This paper discussed the detection of pseudo-random sequence generated by the RC4 algorithm. The detection was achieved by combining the NIST detection methods and visual methods. The visualization results became more abundant by changing the parameters. The research result shows that using different key lengths will result in different results and that longer key length will lead to more aggregated image.

Keywords

Stream Cipher, Pseudo-Random Sequence, Visualization, Randomness Testing

基于块内频数检测法对RC4算法产生的伪随机序列检测

陈河源[1]，郑智捷[2]

[1]云南大学软件学院信息安全系，云南 昆明

[2]云南省软件工程重点实验室，云南 昆明
Email: heyuanchen91@163.com

收稿日期：2015年5月19日；录用日期：2015年6月6日；发布日期：2015年6月10日

摘 要

序列密码算法的安全性主要取决于算法所产生的伪随机序列的随机性[1]。NIST给出16种伪随机序列检测方法[2]，但它们的适用范围是对选定的一段序列进行检测。如何对大量数据段随机性进行检测还需要系统化探讨。本文结合NIST的伪随机序列检测法辅以可视化方法，并对RC4算法产生的伪随机序列进行随机性检测。通过改变测量参数使可视化的结果更为丰富[3]。通过比较分析，观察到使用不同的密钥长度所得到的图像结果不同，密钥越长特征图像的点聚集程度越高。

关键词

序列密码，伪随机序列，可视化，随机性检测

1. 引言

密码学是信息安全领域中最为重要的分支[4]。通过加密手段传输数据已是我们不得不做的事情。对于不断提升的硬件水平，原来我们认为安全、不可攻破的加密算法和密钥长度已经不能保证安全。这迫使我们不断的研发新的加密算法。序列密码因其加解密速度快，很好的适应了现在信息高速传输的时代。

RC4 加密算法是 Ronald Rivest 在 1987 年设计的密钥长度可变的序列密码算法[5]。RC4 算法的加解密速度大约是 DES 算法的 10 倍左右，是最广泛使用的序列密码算法。

本文借助 NIST 中的随机序列检测方法对 RC4 算法的密钥流进行测算、绘图、比较。在对 RC4 算法密钥流的绘图过程中发现改变 RC4 算法的密钥长度，密钥越长图像越汇集。这对序列密码研究中密钥流的研究具有一定的价值。

2. 系统体系构架

本文借助 NIST 的随机序列检测方法可以直观的得到一段数据流随机性的相关数据 Pvalue，可以便利的用其与随机序列的 Pvalue 进行比较，判断其是否是随机的。本文主要采用了块内频数检验法，计算 M 位块中“1”和“0”数比例的分布情况[6]。对所得到的大量测算结果进行可视化，从可视化的结果对密钥流的随机性进行分析。

本文所做的研究主要步骤有：数据流预处理、对数据流进行分段处理、计算 Pvalue、统计不同 Pvalue 的数据段个数、通过固定的绘图公式进行可视化。体系图如图 1 所示。

数据流预处理部分主要是将加密算法所产生的密钥流转化为二进制序列[7]，以便于采用 NIST 的随机数检测方法进行计算统计；由于所处理的数据并不是常规的几百几千位的数据，而是大量超长的数据段，所以需要将数据流进行一个分段，在分段之后再进行 Pvalue 的测算；统计数据段个数部分的意义在于不同的序列的随机性虽然不同，但是当序列段的数目足够大时，就会得到 Pvalue 计算结果相同的数据段，我们将这些数据段分类处理更容易得到进一步的结果；在得到不同 Pvalue 的数据段的数目后我们需要一定的方法将我们所得到的结果进行可视化操作以便于我们进行分析，否则面对大量的数据我们无从

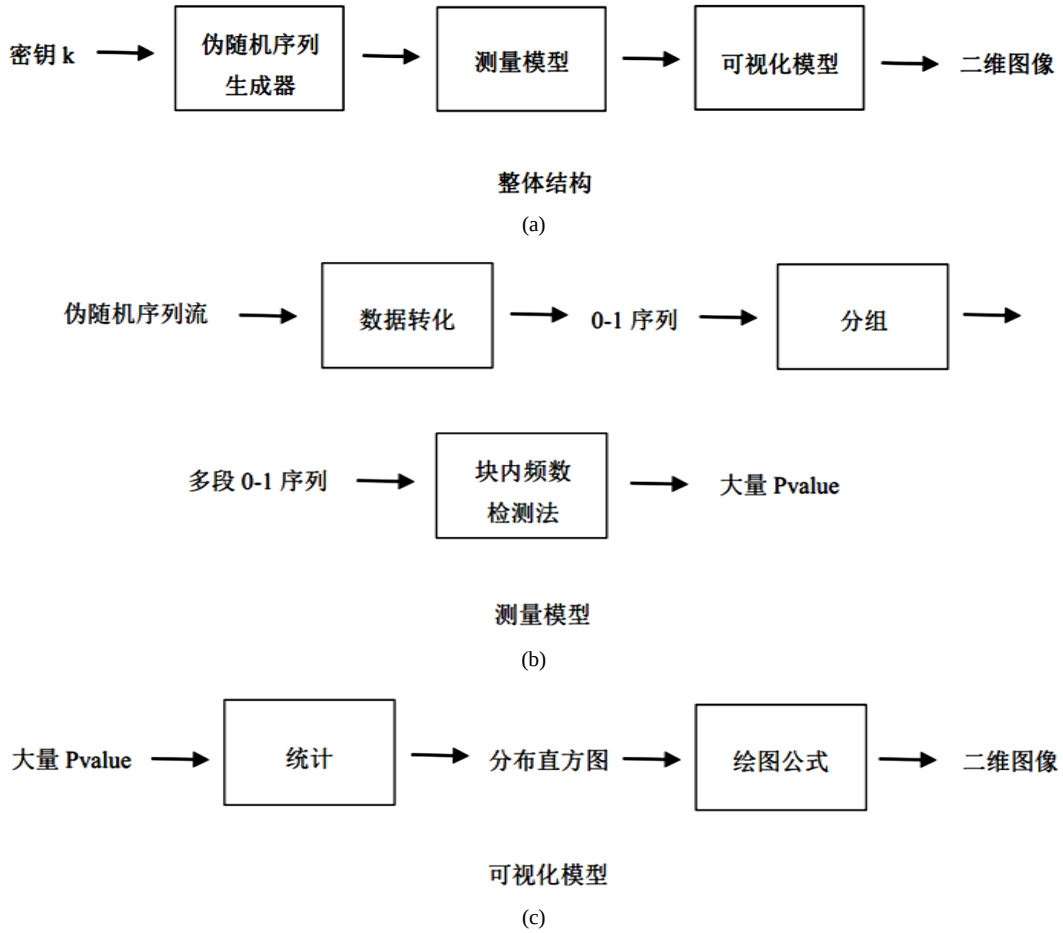

Figure 1. System architecture
图 1. 系统体系结构

着手分析。在得到可视化结果后可以让研究人员直观的观察数据的分布情况，便于总结出图形的固定特征。

本文为伪随机序列的检测从数学计算及可视化角度给出了一种综合研究分析方法，尝试性的打开科学且便利的伪随机序列检测方法的大门。

3. 核心模块方法

3.1. 数据流预处理

NIST 所给出的随机序列测量方法针对的是一串连续的 0-1 序列，所以需要先将序列密码算法所产生的伪随机序列转换为一串连续的二进制 0-1 序列，再作为 NIST 随机性检测方法的输入数据[8]。

3.2. 分组

NIST 所给出的随机序列检测方法虽然没有明确指出所测量数据段的建议长度，不过在一段有特定含义的数据段达到一定的长度后，在某些指标上来看，它即将变为相对随机的数据段。所以我们尝试将数据流分段测量，分段后再运用 NIST 的随机序列检测方法进行 Pvalue 的计算。

我们假设初始序列长度为 N，先把序列分为长度为 L 的数据段，不足 L 舍弃，则共得到[N/L]段数据

段[9]，我们可以在图上画[N/L]个点；对新得到的长度为 L 的数据段在进行一次分组，把序列分为长度为 S 的数据段，不足 S 舍弃，则对于长度为 L 的数据段共得到[L/S]段数据段。所以初始的长度为 N 的序列共被我们分成了[N/L]*[L/S]段数据段，我们通过计算得到[N/L]*[L/S]个 Pvalue。

3.3. 计算 Pvalue

NIST 给出了 16 中随机序列检测方法，不同的检测方法所针对的检测目的不同[10]。通过不同方法所计算出的 Pvaule 可以从多个方面衡量一个序列的随机性。本文对分段以后的数据进行了 Pvalue 的计算，选取了块内频数检测法检测 M 位块中“0”和“1”所占的比例。

3.4. 统计

在所测量的密钥流足够的长的情况下，通过分段会得到海量的 Pvalue。庞大的数据量已无法用传统的分析方法进行分析，所以本文对所得到的 Pvalue 进行数量统计，将相同的 Pvalue 数据段归为一类[11]。在分段后得到长为 S 位的分段，每段可以得到一个计算出的 Pvalue。统计示意图如图 2 所示。

在密钥流足够长的情况下，所得到的 Pvalue 的情况也是各式各样的，实际情况比图 2 的情况又复杂许多，所以我们还需要进行进一步的统计。统计 Pvalue 取 0 ~ 0.1、 0.1 ~ 0.2、 0.2 ~ 0.3、……0.9 ~ 1.0 的数据段的个数。再计算出 Pvalue 取 0 ~ 0.1、 0.1 ~ 0.2、 0.2 ~ 0.3、……0.9 ~ 1.0 时各组所占的百分比，分别记为 p1, p2, p3……p10 [12]。

3.5. 生成可视化图像

在得到分类整理好的统计数据之后，我们需要将统计数据很好的表达在图像上，本文所采取的是二维图像的表示方法[13]。绘图公式如下所示(每 L 位长的数据在图像上打一个点)：

$$令X=\sum_{i=1}^{10} pi^{t}$$

$$令Y=\sum_{i=1}^{10} \sqrt[t']{pi}$$

$$\left(其中\sum_{i=1}^{10} pi=1\right)$$

通过以上绘图公式，我们可以得到的图像中如图 3 所示。

3.6. 测量参数的改变

本文中的可视化模型共涉及到四个可以改变图形的参数，即打点参数 t 和 t’，分段参数 L 和 S。从绘图公式中我们可以根据指数函数图像直观的估计改变参数 t 和 t’所带来的图像变化，所以在后面我们进行测量、可视化的过程中我们优先改变 t 和 t’。选定了合适的 t 和 t’之后，我们再改变分段参数 L 和 S 来观察图像的变化。参数的可选范围和所选范围如表 1 所示。

Table 1. Optional scope and selected scope of the parameters
表 1. 参数可选范围及所选范围

	t	t’	L	S
可选范围	1, 2, 3…	1, 2, 3…	1, 2, 3…	1, 2, 3…
所选范围	1, 2, 3, 4	1, 2, 3, 4	160 - 28,800	20 - 480

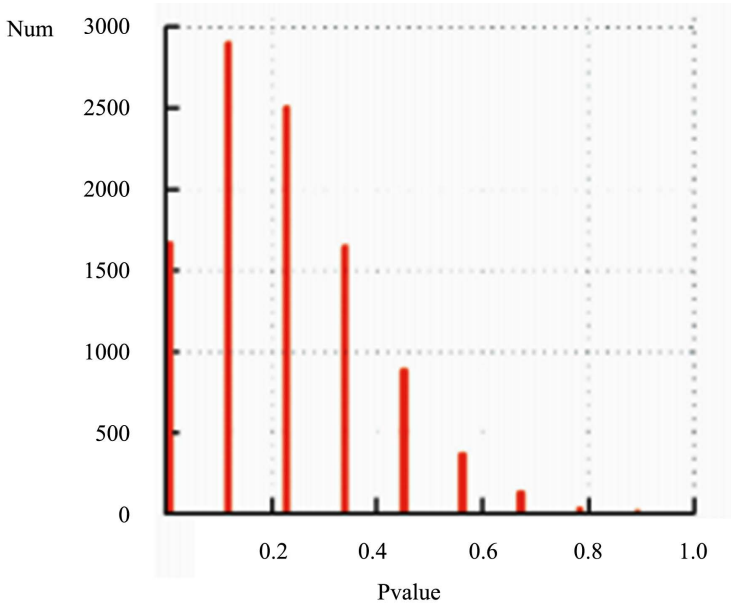

Figure 2. Statistics on the number of Pvalue
图 2. 对 Pvalue 数量的统计

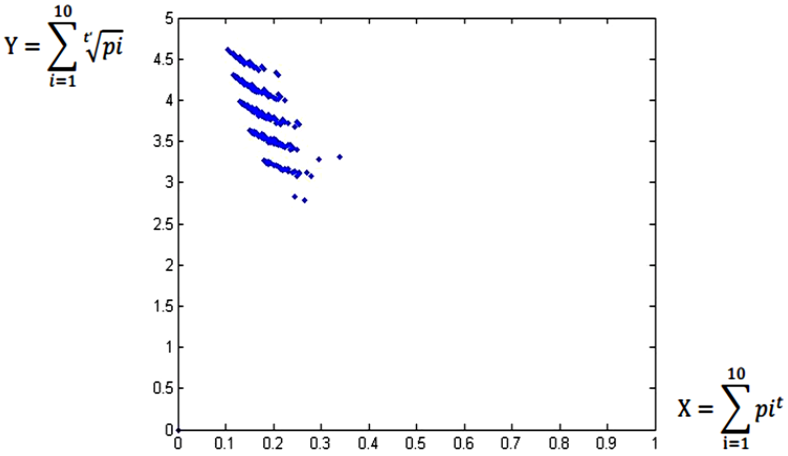

Figure 3. Comp image using drawing formula
图 3. 利用绘图公式所得到的样图

4. 测量结果

本文利用上述提到的模块方法对 RC4 算法所产生的密钥流进行系统化的检测，挑选其中部分特征明显的结果进行展示。在对 RC4 算法改变密钥长度进行绘图检测之前，本文所采用的 RC4 密钥长度为 128bit，即一般情况下的最短密钥长度，使得改变参数 t、t’、L 和 S 的结果是可信的伪随机序列的检测结果。本文采用了控制变量法，通过一次改变一个相关变量来观察图像的变化趋势。本文优先选择了改变参数 t 和 t’来观察图像的变化，因为通过观察绘图公式易知改变参数 t 和 t’对图像所带来的改变，在改变参数 t 和 t’时，在一个矩阵中纵向改变 t 横向改变 t’，使得 t 和 t’的综合变化结果更易于观察。其次，再尝试改变参数 L 和参数 S，在改变参数 L 和参数 S 时，逐步加大变化的程度，呈现出渐变和极端的情况。最后探究了改变 RC4 算法的密钥长度对可视化结果的影响，最短采用的是 1 字节的密钥，最长采用的是 16

字节的密钥。

5. 结果分析

从图 4 中我们可以看到在改变参数 t 和 t’的图形变化趋势：增大 t，图形向 y 轴靠近，增大 t’图形慢慢远离 x 轴。这点从观察绘图公式即可比较容易理解，t 是 pi 的 t 次方，p 作为概率值是一个介于 0 和 1 之间的数字，由指数函数的图像可知增大 t 则 pi 的值变小，则图形向 y 轴靠近；t’是 pi 的开 t 次方，由于 p 是一个小于 1 大于 0 的数字，由指数函数的图像可知增大 t’则 pi 的值变大，则图形远离 x 轴。图 4 更大的意义或许在于帮助我们选择一个比较合适的 t 和 t’为后面改变其他参数做铺垫。由于横向是增大 t’，纵向是增大 t，通过这个矩阵我们可以直观的看到同时增大 t 和 t’，同时减小 t 和 t’，单独增大 t，单独增大 t’等一系列的情况。最后本文选择了 t = 2，t’ = 3 的情况来展开后面的研究。

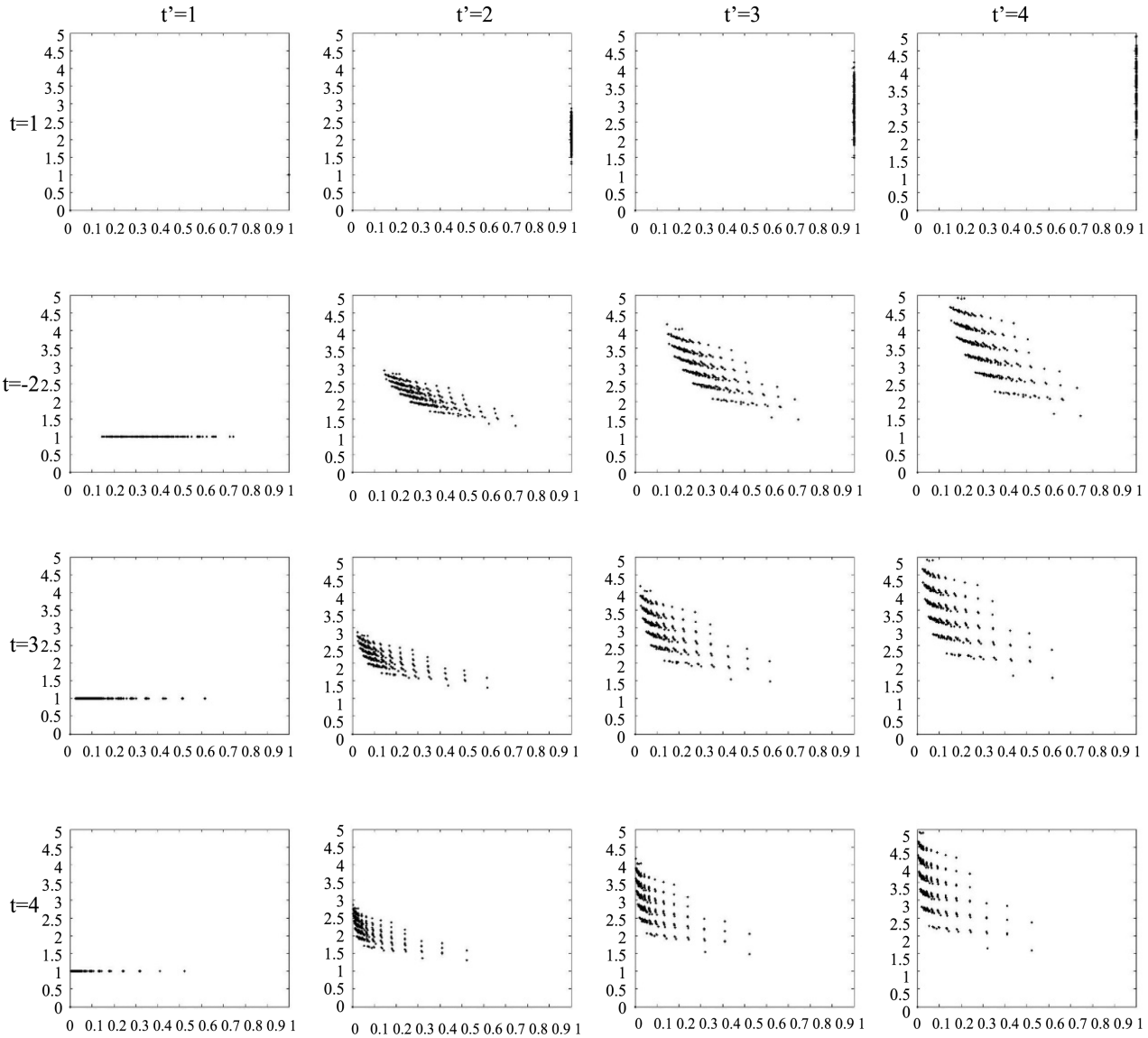

Figure 4. Frequency test within a block with L = 3200 S = 160
图 4. 块内频数检测法 L = 3200 S = 160

从图 5 中我们可以看到改变参数 L 的图形变化趋势：增大 L，图像的点逐渐增多并且重合的点也逐渐增多，图像的分层逐渐变得明显最后又汇聚到一起。

从图 6 中我们可以看到改变参数 S 的图形变化趋势：增大 S，图像逐渐开始分层最后变得越来越稀疏，图像的点在减少的同时并且重合的点也逐渐减少。

经由对图 5 和图 6 的比较，我们不难发现在块内频数检测法下增大 L 与缩小 S 效果近似，都是一个图像逐渐分层再汇聚的过程。

从图 7 中我们可以看到在 RC4 算法分别采用 1 至 16 字节密钥的情况下所产生的密钥流在块内频数检测法下 L = 3200，S = 80，t = 2，t’ = 3 的检测结果。我们知道 RC4 算法在一般情况下密钥长度超过 128 bit 认为是安全的，所以我们可以多留意采用 16 字节密钥的图像情况。从图像中看到增加密钥长度图像是一点一点变化的，总体上看是一个汇集的过程，但是在汇集的过程中更长密钥的图像也有轻微扩散的情况。

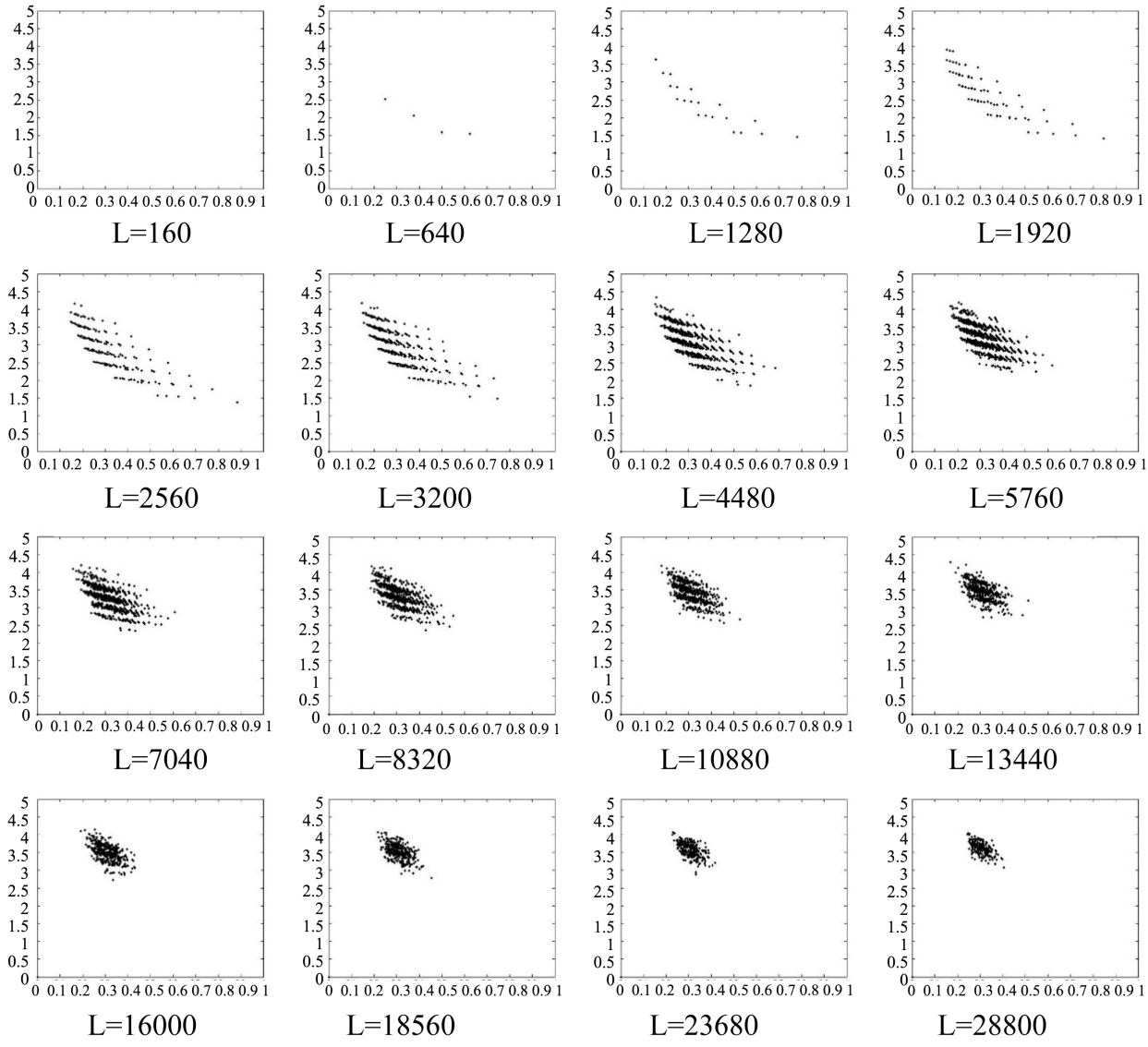

Figure 5. Frequency test within a block with S = 160 t = 2 t’ = 3
图 5. 块内频数检测法 S = 160 t = 2 t = 3

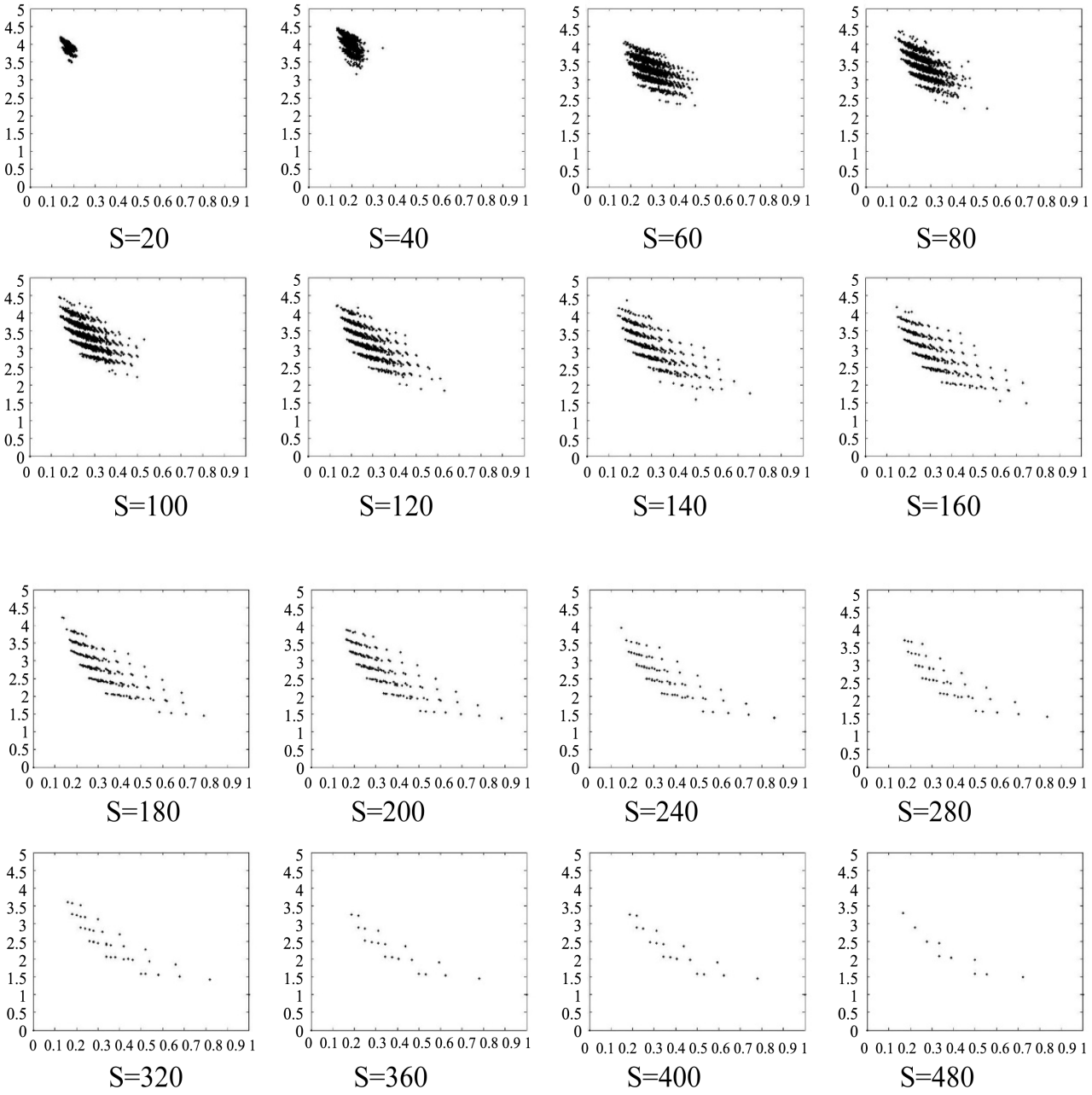

Figure 6. Frequency test within a block with L = 3200 t = 2 t’ = 3
图 6. 块内频数检测法 L = 3200 t = 2 t’ = 3

比较相邻的两幅区别不大，但比较密钥长度差距大的（如密钥为 1 字节和密钥为 16 字节）图像差距较为明显。由此我们猜测对于 RC4 算法在密钥长度逐渐增加的过程中，密钥流的随机性逐渐变好。这样的结果也与我们所知道的加密算法的特点相一致。

致谢

感谢国家自然科学基金、云南大学软件学院以及云南省软件工程重点实验室对信息安全研究项目的基金支持。

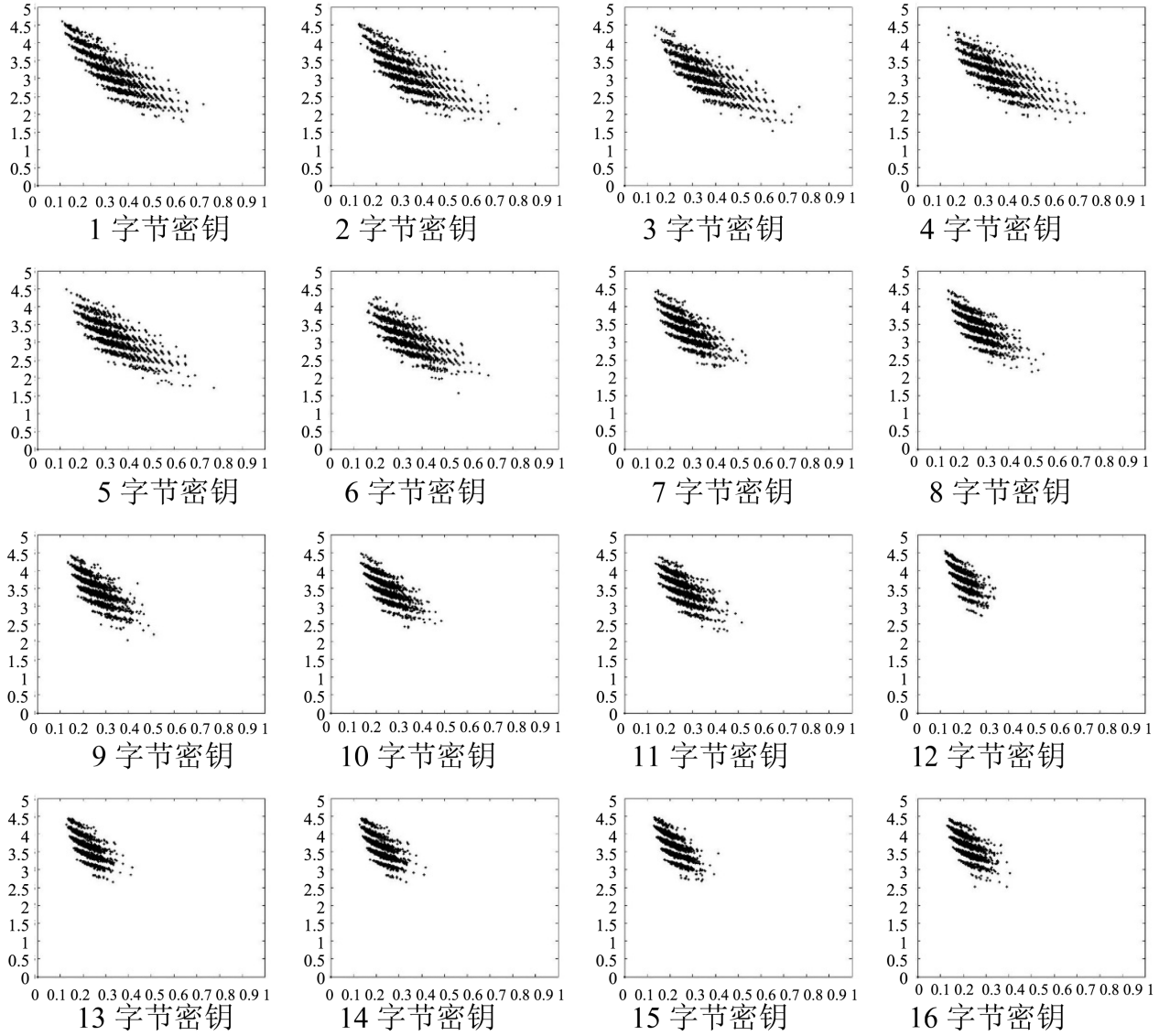

Figure 7. Changing the key length of RC4 algorithm, frequency test within a block with L = 3200 S = 80
图 7. RC4 改变密钥长度采用块内频数检测法 L = 3200 S = 80

基金项目

国家自然科学基金资助项目(61362014)。

参考文献 (References)

[1] 刘建夏 (2005) 一种混沌伪随机序列的设计及其应用. *计算机工程*, **18**, 150-152.

[2] NIST (2010) A statistical test suite for random and pseudorandom number generators for cryptographic applications. http://csrc.nist.gov/groups/ST/toolkit/rng/documents/SP800-22rev1a.pdf

[3] 张巍琼, 郑智捷 (2012) 基于不同产生机制的伪随机序列和 DNA 序列的随机性测量. *成都信息工程学院学报*, **6**, 文章编号: 1671.

[4] 邓绍江 (2005) 混沌理论及其在信息安全中的应用研究. 博士论文, 重庆大学, 重庆.

[5] 赵伟, 曹云飞 (2013) RC4 的密钥碰撞. *通信技术*, **12**, 74-76.

[6] 朱小兵 (2012) 基于统计随机性的 Hash 函数安全评估模型研究. 硕士论文, 西南交通大学, 成都.

[7] 周垚 (2013) 基于多维可视化的动态生成序列测量体系. 硕士论文, 云南大学, 昆明

[8] 张美玲 (2010) 密码算法测试平台. 硕士论文, 西安电子科技大学, 西安.

[9] 赵建秀, 王洪国, 邵增珍, 张岳, 丁艳辉 (2013) 一种基于信息熵的时间序列分段线性表示方法. *计算机应用研究*, **8**, 2391-2394.

[10] 苏桂平, 刘争春, 姚旭初, 殷学文 (2006) 一种信息安全系统中序列随机性检验方法. *计算机工程*, **8**, 210-215.

[11] Zheng, J.Z.J., Zheng, C.H.H. and Kunii, T.L. (2011) A Framework of Variant Logic Construction for Cellular Automata. http://www.intechopen.com/books/cellular-automata-innovative-modelling-for-science-and-engineering/a-framework-of-variant-logic-construction-for-cellular-automata

[12] Zheng, J.Z.J and Zheng, C.H. (2010) A framework to express variant and invariant functional spaces for binary logic. *Frontiers of Electrical and Electronic Engineering in China*, **5**, 163-172.

[13] Li, Q.P. and Zheng, J. (2010) 2D Spatial Distributions for Measures of Random Sequences Using Conjugate Maps. *The* 11*th Australian Information Warfare and Security Conference*, Perth, 30 November-2 December 2010, 18-25.