Übersetzte Version von SP800-22rev1.txt

Special Publication 800-22
Revision 1

Ein statistischer Test Suite für Zufällige und Pseudorandom Number Generators für Cryptographic Applications

AndrewRukhin, JuanSoto, JamesNechvatal, Miles Smid, ElaineBarker, StefanLeigh, MarkLevenson, Mark Vangel, DavidBanks, AlanHeckert, JamesDray, San Vo

Überarbeitet: August2008
LawrenceEBasshamIII

NIST Special Publication 800-22
Revision 1
ATION 800-22
Revision 1

Ein statistischer Test Suite für die Zufalls-und Pseudo-Zufallszahlen-Generatoren für Cryptographic Applications

Andrew Rukhin1, Juan Soto2, James Nechvatal2, Miles Smid2, Elaine Barker2, Stefan Leigh1, Mark Levenson1, Mark Vangel1, David Banken1, Alan Heckert1, James Dray2, San Vo2

Überarbeitet: August 2008
Lawrence E Bassham III2

COMP UTER SECURITY

1Statistical Technik Division2Computer Security Division
Laboratorium für Informationstechnologie
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

Überarbeitet: August 2008

U. S. Department of Commerce

Carlos M. Gutierrez, Secretary

National Institute of Standards and Technology

James M. Turner, stellvertretender Direktor

Analyse

Berichte über Computer Systems Technologie

Die Information Technology Laboratory (ITL) am National Institute of Standards and Technology
(NIST) fördert die US-Wirtschaft und Gemeinwohl, indem sie technische Führung für die Nation
Mess-und Normen-Infrastruktur. ITL entwickelt Tests, Testmethoden, Referenzdaten, Nachweis der
Konzept-Implementierungen und die technische Analyse, die Entwicklung und den produktiven Einsatz des Vorschusses
Informationstechnologie. ITL seinen Aufgaben zählt die Entwicklung von technischen, physikalischen,
Verwaltungs-und Management-Standards und Richtlinien für die kostengünstige Sicherheit und Privatsphäre
sensible Informationen klassifiziert in Federal Computer-Systeme. Dieses Special Publication 800-Serie
Berichte über ITL-Forschung, Beratung und aufsuchende Bemühungen in Computer-Sicherheit und ihre Zusammenarbeit
Aktivitäten mit der Industrie, Regierung und akademischer Einrichtungen.

National Institute of Standards and Technology Special Publication 800-22 Revision 1
Natl. Inst. Stand. Technol. Spec. Publ. 800-22rev1, 131 Seiten (August 2008)

Bestimmte gewerbliche Einrichtungen, Geräten oder Materialien können in dieser identifiziert werden
Dokument um ein experimentelles Verfahren oder Konzept adäquat zu beschreiben.
Eine solche Identifizierung ist nicht beabsichtigt, Empfehlung oder Billigung durch die implizieren
National Institute of Standards and Technology, noch ist es zu verstehen, dass die Personen, Materialien oder Geräte sind unbedingt die beste für diesen Zweck verfügbar.

Analyse

Table of Contents

1

Analyse

v

Analyse

Abstrakt
Dieses Papier beschäftigt sich mit einigen Aspekten der Auswahl und Prüfung zufällig und Pseudo-Zufallszahlen-Generatoren.
Die Ausgänge dieser Generatoren können in vielen kryptographischen Anwendungen eingesetzt werden, wie die Erzeugung von
Schlüssel Material. Generatoren für den Einsatz in kryptographischen Anwendungen müssen stärker treffen
Anforderungen als für andere Anwendungen. Insbesondere müssen die Ausgänge werden in Abwesenheit unberechenbar
Wissen über die Eingänge. Einige Kriterien für die Charakterisierung und Auswahl geeigneter Generatoren sind
diskutiert in diesem Dokument. Das Thema der statistischen Tests und ihre Beziehung zur Kryptoanalyse ist auch
diskutiert, und einige empfohlene statistische Tests zur Verfügung. Diese Tests können sinnvoll sein, als ein erster Schritt
in der Entscheidung, ob ein Generator für eine bestimmte kryptographische Anwendung ist. Allerdings
keine Reihe von statistischen Tests kann absolut bestätigen, ein Generator als geeignet für den Einsatz in einem bestimmten
Anwendung, dh, können statistische Tests nicht als Ersatz für Kryptoanalyse dienen. Das Design und die
Kryptoanalyse von Generatoren liegt außerhalb des Rahmens dieser Arbeit.

Key words: Zufallszahlengenerator Hypothese zu testen, P-Wert

Zusammenfassung-1

Analyse

1. Introduction to Random Number Testing
Die Notwendigkeit für zufällige und Pseudo-Zufallszahlen entsteht in vielen kryptographischen Anwendungen. Für
Beispielsweise verwenden gemeinsame Kryptosysteme Tasten, die nach dem Zufallsprinzip generiert werden müssen. Viele
kryptographische Protokolle erfordern auch Zufalls-oder Pseudo-Eingänge an verschiedenen Stellen, zB für Hilfs-
Mengen bei der Erzeugung digitaler Signaturen verwendet werden, oder zur Erzeugung von Herausforderungen in Authentifizierungsprotokolle.

Dieses Dokument beschreibt die Zufälligkeit Prüfung von Zufallszahlen und Pseudo-Zufallszahlen
Generatoren, die für viele Zwecke verwendet werden können, einschließlich kryptographische, Modellierung und Simulation
Anwendungen. Der Schwerpunkt dieses Dokuments ist auf Anwendungen, bei denen Zufall für erforderlich
kryptographische Zwecke. Eine Reihe von statistischen Tests auf Zufälligkeit ist in

diesem Dokument beschrieben. Die
National Institute of Standards and Technology (NIST) glaubt, dass diese Verfahren nützlich sind
Erkennen Abweichungen eine binäre Folge von Zufälligkeit. Allerdings sollte ein Tester fest, dass offensichtlich
Abweichungen von Zufälligkeit kann auf entweder eine schlecht konzipierte Generator oder um Anomalien, die angezeigt werden
in den binären Sequenz, die getestet (dh eine bestimmte Anzahl von Fehlern ist in zufälligen Sequenzen zu erwarten ist
produziert von einem bestimmten Generator). Es liegt an den Tester auf die richtige Interpretation des Tests bestimmen
Resultate. Siehe Abschnitt 4 für eine Diskussion der Prüfstrategie und die Interpretation der Testergebnisse.

1.1 Allgemeine Diskussion
Es gibt zwei grundlegende Arten von Generatoren verwendet, um zufälligen Sequenzen zu erzeugen: Zufallszahlengeneratoren
(RNGs-siehe Abschnitt 1.1.3) und Pseudo-Zufallszahlen-Generatoren (PRNGs-siehe Abschnitt 1.1.4). Für
kryptographische Anwendungen, produzieren diese beiden Typen Generator einen Strom von Nullen und Einsen, dass sein
unterteilt in Teilströme oder Blöcken von Zufallszahlen.

1.1.1 Randomness
Eine zufällige Bitfolge könnte als das Ergebnis der Flips von einem unvoreingenommenen "fair" Münze mit Seiten interpretiert werden
, die "0" und "1", mit jedem Flip mit einer Wahrscheinlichkeit von exakt beschriftet ½ der Herstellung einer "0" oder "1".
Darüber hinaus die Flips sind unabhängig von einander: das Ergebnis einer früheren Münzwurf hat keinen Einfluss auf
Zukunft Münzwürfe. Der unvoreingenommene "fair" Münze ist somit die perfekte zufälligen Bitstrom-Generator, da die "0"
und "1"-Werte werden nach dem Zufallsprinzip verteilt werden (und [0,1] gleichmäßig verteilt). Alle Elemente des
Sequenz sind unabhängig voneinander erzeugt und den Wert des nächsten Elements in der Reihenfolge
lässt sich nicht vorhersagen, unabhängig davon, wie viele Elemente wurden bereits hergestellt werden.

Offensichtlich ist die Verwendung von unvoreingenommene Münzen für kryptographische Zwecke unpraktisch. Dennoch ist die
hypothetischen Ausgang einer solchen idealisierten Generator einer echten Zufallsfolge dient als Benchmark für
die Bewertung der zufälligen und Pseudo-Zufallszahlen-Generatoren.

1.1.2 Unvorhersehbarkeit
Zufällige und Pseudo-Zufallszahlen für kryptographische Anwendungen erzeugt werden soll unvorhersehbar.

Im Falle der PRNGs, wenn der Samen ist nicht bekannt, sollte die nächste Ausgabe Zahl in der Folge werden
unvorhersehbare trotz aller Kenntnisse der bisherigen Zufallszahlen in der Sequenz. Diese Eigenschaft ist
bekannt als Vorwärts Unberechenbarkeit. Es sollte auch nicht möglich sein, den Samen von Wissen bestimmen
der generierten Werte (dh rückwärts Unberechenbarkeit ist auch erforderlich). Keine Korrelation zwischen einem Samen
und jeder Wert aus, dass das Saatgut erzeugt werden soll evident, jedes Element der Sequenz sollte erscheinen
werden die Ergebnisse einer unabhängigen zufälligen Ereignis, dessen Wahrscheinlichkeit ist 1 / 2.

Um sicherzustellen, freuen Unberechenbarkeit ist darauf zu achten bei der Beschaffung von Saatgut ausgeübt werden. Die Werte von einer produzierten
PRNG sind völlig vorhersagbar, wenn das Saatgut und Generation-Algorithmus bekannt sind. Da in vielen Fällen

Analyse

die Generation Algorithmus ist öffentlich zugänglich, muss die Saat geheim gehalten werden und sollten nicht ableitbar
aus dem Pseudo-Sequenz, die sie produziert. Darüber hinaus muss das Saatgut selbst unberechenbar.

1.1.3 Random Number Generators (RNG)
Die erste Art von Sequenz-Generator ist ein Zufallsgenerator (RNG). Ein RNG verwendet eine deterministische
Quelle (dh, die Entropie-Quelle), zusammen mit einigen Verarbeitung Funktion (dh die Entropie
Destillation) zu Zufälligkeit zu erzeugen. Die Verwendung einer Destillation benötigt, um überhaupt zu überwinden
Schwäche in den Entropie-Quelle, dass die Ergebnisse in die Produktion von nicht-zufällige Zahlen (zB das Auftreten
von einer langen Kette von Nullen oder Einsen). Die Entropie-Quelle besteht typischerweise aus einigen physikalischen Größe, wie
den Lärm in einer elektrischen Schaltung, das Timing der User-Prozesse (zB Tastenanschläge oder Mausbewegungen), oder
Quanteneffekte in einem Halbleiter. Verschiedene Kombinationen dieser Eingänge verwendet werden.

Die Ausgänge eines RNG kann direkt als Zufallszahl verwendet werden oder können in eine Pseudo-zugeführt werden
Number Generator (PRNG). Um direkt verwendet werden (dh ohne weitere

Verarbeitung), die Ausgabe von RNG
muss strenge Kriterien Zufälligkeit gemessen durch statistische Tests erfüllen, um festzustellen, dass die
physikalischen Ursachen der RNG-Eingänge erscheinen zufällig. Zum Beispiel eine physikalische Quelle wie elektronische
Lärm kann eine Überlagerung von regelmäßigen Strukturen, wie Wellen oder andere periodische Phänomene,
die erscheinen mag zufällig sein, die noch zu bestimmenden nicht-zufällige mittels statistischer Tests.

Für kryptographische Zwecke, muss die Ausgabe von RNGs unvorhersehbar zu sein. Doch einige physikalische
Quellen (z. B. Datum / Zeit-Vektoren) sind ziemlich vorhersehbar. Diese Probleme können durch die Kombination gemildert werden
Ausgänge aus verschiedenen Quellen, die als Eingänge für eine RNG verwenden. Doch die daraus resultierenden Ergebnisse
von der RNG kann noch mangelhaft, wenn sie von statistischen Tests ausgewertet. Darüber hinaus die Produktion von
hochwertige Zufallszahlen kann zu zeitaufwändig, so dass eine solche Produktion unerwünscht, wenn ein
große Menge von Zufallszahlen benötigt. Um große Mengen von Zufallszahlen zu erzeugen,
Pseudo-Zufallszahlen-Generatoren kann vorteilhaft sein.

1.1.4 Pseudo-Zufallszahlen-Generatoren (PRNGs)
Der zweite Generator-Typ ist ein Pseudo-Zufallszahlen-Generator (PRNG). Ein PRNG verwendet eine oder mehrere
Eingänge und erzeugt multiple "Pseudo"-Nummern. Inputs zu PRNGs heißen Samen. In Kontexten
in die Unvorhersehbarkeit benötigt wird, muss das Saatgut selbst zufällig und unvorhersehbar. Daher wird standardmäßig
ein PRNG sollte die Samen von den Ausgängen eines RNG erhalten, dh erfordert eine PRNG ein RNG als
Begleiter.

Die Ausgänge eines PRNG sind in der Regel deterministische Funktionen des Saatgutes, das heißt, alles echte Zufälligkeit
beschränkt sich auf Seed-Generation. Die deterministische Natur des Prozesses führt zu dem Begriff "Pseudo".
Da jedes Element einer Pseudozufallssequenz reproduzierbar ist aus ihrem Samen, braucht nur den Samen zu sein
gespeichert, wenn Vervielfältigung oder Validierung der Pseudozufallssequenz erforderlich ist.

Ironischerweise Pseudo-Zufallszahlen häufig zu sein scheinen eher zufällig als Zufallszahlen erhalten aus
physischen Quellen. Wenn ein Pseudozufallssequenz richtig aufgebaut ist, wird jeder Wert in der Sequenz

hergestellt aus den vorherigen Wert über Transformationen, die auf zusätzliche Zufälligkeit einführen erscheinen. A
Reihe solcher Transformationen können eliminieren statistische Auto-Korrelationen zwischen Eingang und Ausgang. So
die Ausgänge eines PRNG haben bessere statistische Eigenschaften und schneller als ein Zufallsgenerator erzeugt werden.

1.1.5 Testing
Verschiedene statistische Tests können zu einer Sequenz angewendet werden, um zu versuchen, zu vergleichen und bewerten die Sequenz zu einem
wirklich zufälliger Reihenfolge. Der Zufall ist ein probabilistischer Eigentum, das heißt, die Eigenschaften eines zufälligen

Analyse

Sequenz kann charakterisiert und beschrieben werden in Form von Wahrscheinlichkeiten. Das wahrscheinliche Ergebnis der statistischen
Tests, wenn ein wirklich zufälliger Reihenfolge angewendet, ist a priori bekannt und können in probabilistischen beschrieben werden
Bedingungen. Es gibt eine unendliche Anzahl möglicher statistischer Tests, die jeweils die Bewertung der Anwesenheit oder Abwesenheit von
ein "Muster", die, wenn erkannt wird, zeigen, dass die Reihenfolge nicht zufällig ist, wäre. Da gibt es so viele
Tests für die Beurteilung, ob eine Sequenz zufälliger oder nicht, ist keine spezielle endliche Menge von Tests als "komplett".
Darüber hinaus müssen die Ergebnisse der statistischen Tests mit einiger Vorsicht interpretiert werden und Vorsicht zur Vermeidung
falschen Schlüssen über einen bestimmten Generator (siehe Abschnitt 4).

Ein statistischer Test wurde entwickelt, um eine bestimmte Nullhypothese (H0) zu testen. Für die Zwecke dieses Dokuments,
die Null-Hypothese unter Test ist, dass die Reihenfolge getestet zufällig ist. Verbunden mit dieser null
Hypothese ist die Alternativhypothese (Ha), die für dieses Dokument ist, dass die Reihenfolge nicht
zufällig. Für jede Anwendung testen, ist eine Entscheidung oder Schlussfolgerung ableiten, dass akzeptiert oder verwirft die null
Hypothese, dh ob der Generator (oder nicht) produzieren zufällige Werte, basierend auf der Sequenz,
produziert wurde.

Für jeden Test muss eine entsprechende Zufälligkeit Statistik ausgewählt und verwendet werden, um die Akzeptanz zu ermitteln oder
Ablehnung der Nullhypothese. Unter der Annahme, des Zufalls, hat eine solche Statistik

eine Verteilung von
möglichen Werte. Eine theoretische Bezug Verteilung dieser Statistik unter der Nullhypothese
bestimmt durch mathematische Methoden. Von diesem Verweis Vertrieb, ist ein kritischer Wert ermittelt
(In der Regel ist dieser Wert "far out" in die Ränder der Verteilung, sagen bei der 99%-Punkt). Bei einem Test, einem
Teststatistik Wert wird auf die Daten (die Abfolge im Test) berechnet. Diese Teststatistik-Wert ist
im Vergleich zu den kritischen Wert. Wenn die Teststatistik-Wert den kritischen Wert überschreitet, wird die Nullhypothese für
Zufälligkeit wird abgelehnt. Andernfalls wird die Nullhypothese (die Zufälligkeit Hypothese) nicht verworfen (dh
die Hypothese wird akzeptiert).

In der Praxis ist der Grund, dass statistische Hypothesentests arbeitet, dass der Verweis Verteilung und die
kritischen Wert abhängig sind und generiert unter eine vorläufige Annahme der Zufälligkeit. Wenn der
Zufälligkeit Annahme ist in der Tat gilt für die Daten zur Hand, dann die daraus resultierenden berechneten Testgröße
Wert auf die Daten haben eine sehr geringe Wahrscheinlichkeit (zB 0,01%) über dem kritischen Wert.
Auf der anderen Seite, wenn die berechnete Teststatistik-Wert nicht übersteigt den kritischen Wert (dh, wenn die niedrigen
Wahrscheinlichkeit Ereignis wird in der Tat vorkommen), dann aus einem statistischen Hypothesentest Sicht der niedrigen
Wahrscheinlichkeit Ereignis sollte der Natur nicht vor. Deshalb, wenn die berechnete Teststatistik Wert überschreitet
den kritischen Wert, ist der Abschluss gemacht, dass die ursprüngliche Annahme der Zufälligkeit verdächtigen oder fehlerhaft ist.
In diesem Fall liefert statistische Hypothesentests die folgenden Schlussfolgerungen: H0 verwerfen (Zufälligkeit) und
akzeptieren Ha (non-Zufälligkeit).

Statistische Hypothesentests ist ein Abschluss-Generation Prozedur, die zwei mögliche Ergebnisse hat, entweder
akzeptieren H0 (die Daten zufällig) oder akzeptieren Ha (die Daten nicht zufällig). Die folgenden 2 by 2 Tabelle bezieht sich
den wahren (unbekannten) Status der Daten zur Hand, um den Abschluss kamen mit dem Testverfahren.

Wahre Situation
FAZIT
Accept Accept H0 Ha (H0 verwerfen)
Die Daten werden random (H0 wahr ist) ist kein Fehler vom Typ I-Fehler
Die Daten werden nicht random (Ha ist true) Type II error Kein Fehler

Wenn die Daten, in Wahrheit, zufällig, dann ein Rückschluss auf die Nullhypothese (dh dem Schluss, dass die Daten
ist nicht zufällig) wird ein kleiner Prozentsatz der Zeit auftreten. Diese Schlussfolgerung wird als Typ-I-Fehler. Wenn der
Daten ist in Wahrheit, nicht zufällig, dann ein Rückschluss auf die Null-Hypothese (dh zu akzeptieren, schließen, dass die Daten
wirklich zufällig) wird als Typ-II-Fehler. Die Schlussfolgerungen zu akzeptieren, H0, wenn die Daten wirklich zufällig ist,
und H0 ablehnen, wenn die Daten nicht zufällig ist, richtig sind.

1-3

Analyse

Die Wahrscheinlichkeit für einen Typ-I-Fehler wird oft als das Signifikanzniveau des Tests. Diese Wahrscheinlichkeit kann
vor einer Prüfung eingestellt werden und wird als a bezeichnet Für den Test wird eine ist die Wahrscheinlichkeit, dass der Test zeigen, dass die
Reihenfolge ist nicht zufällig, wenn es wirklich zufällig ist. Das ist, erscheint eine Sequenz zu haben non-random
Eigenschaften, auch wenn ein "guter" Generator die Sequenz erzeugt. Gemeinsame Werte eines
in der Kryptographie
sind etwa 0,01.

Die Wahrscheinlichkeit für einen Typ-II-Fehler wird als ß bezeichnet. Für den Test ß ist die Wahrscheinlichkeit, dass der Test
zeigen, dass die Reihenfolge zufällig ist, wenn es nicht, das heißt, produziert eine "schlechte" Generator eine Sequenz, die
erscheint, um zufällige Eigenschaften haben. Im Gegensatz zu a, ß
ist kein fester Wert. ß
kann auf viele verschiedene Werte annehmen
denn es gibt eine unendliche Anzahl von Möglichkeiten, wie ein Datenstrom
nicht-zufällige können, und jeder anderen
So ergibt sich eine unterschiedliche ß. Die Berechnung der Type II error ß
ist schwieriger als die Berechnung einer
wegen der vielen möglichen Arten von Nicht-Zufälligkeit.

Eines der primären Ziele der folgenden Tests ist die Wahrscheinlichkeit für einen Typ-II-Fehler, dh zu minimieren,
Minimierung der Wahrscheinlichkeit für die Annahme einer Sequenz von einem guten Generator erzeugt, wenn der Generator
war eigentlich schlecht. Die Wahrscheinlichkeiten ein
und ß
sind miteinander und mit der Größe n der getesteten Zusammenhang
Reihenfolge, in der Weise, dass, wenn zwei von ihnen angegeben werden, der dritte Wert automatisch ermittelt wird.

Praktiker in der Regel wählen Sie eine Stichprobengröße n und einen Wert für eine (Die Wahrscheinlichkeit für einen Typ-I-Fehler - das Niveau von Bedeutung). Dann ein kritischer Punkt für eine bestimmte Statistik ausgewählt, die die kleinsten ß produzieren (Der Wahrscheinlichkeit eines Typ-II-Fehler). Das heißt, eine geeignete Stichprobengröße zusammen mit einem akzeptablen ausgewählt Wahrscheinlichkeit zu entscheiden, dass eine schlechte Generator hat die Sequenz erzeugt, wenn es wirklich zufällig ist. Dann wird die Cutoff-Punkt für die Akzeptanz ist so gewählt, dass die Wahrscheinlichkeit eines falschen Annahme einer Sequenz zufälliger hat den kleinsten möglichen Wert.

Jeder Test wird auf einem berechneten Teststatistik-Wert, der eine Funktion der Daten beruht. Wenn die Prüfgröße Wert S und der kritische Wert ist t, dann ist die Typ-I-Fehler Wahrscheinlichkeit P (S> t | | Ho ist wahr) = P (Ho ablehnen | H0 wahr ist), und der Typ II-Fehler Wahrscheinlichkeit P (S = t | | H0 falsch ist) = P (akzeptieren H0 | H0 falsch ist). Der Test Statistik wird ein P-Wert, der die Stärke der Beweise fasst gegen die Null-Berechnung Hypothese. Für diese Tests wird jeder P-Wert ist die Wahrscheinlichkeit, dass ein perfekter Zufallsgenerator würde produziert haben eine Sequenz weniger zufällig als Folge, das getestet wurde, da die Art der nonrandomness beurteilt durch den Test. Wenn ein P-Wert für einen Test ist entschlossen, gleich 1 sein, dann ist die Folge scheint perfekt Zufälligkeit haben. Ein P-Wert von Null bedeutet, dass die Folge zu sein scheint komplett non-random. Ein Signifikanzniveau (a) kann für die Tests ausgewählt werden. Wenn P-Wert = a, dann ist die Nullhypothese akzeptiert wird, dh die Sequenz scheint zufällig zu sein. Wenn P-Wert <a, dann ist die Null-Hypothese wird abgelehnt, dh die Sequenz scheint zu sein nicht zufällig. Der Parameter a bezeichnet die Wahrscheinlichkeit, dass der Typ-I-Fehler. Typischerweise wird ein liegt im Bereich [0,001, 0,01] gewählt.

• Ein a von 0,001 bedeutet, dass man eine Sequenz in 1000 Sequenzen durch abgelehnt werden erwartet der Test, wenn die Sequenz war zufällig. Für einen P-Wert = 0,001, würde eine Sequenz als sein random mit einer Wahrscheinlichkeit von 99,9%. Für einen P-Wert <0,001, würde eine Sequenz als nonrandom werden mit einer Wahrscheinlichkeit von 99,9%.
• Ein a von 0,01 zeigt an, dass man 1-Sequenz in 100 Sequenzen abgelehnt werden erwartet.

Ein pValue
=
0.01 würde bedeuten, dass die Sequenz als wäre zufällig zu sein mit einem Vertrauensniveau von 99%.
Ein P-Wert <0,01 würde bedeuten, dass die Schlussfolgerung war, dass die Reihenfolge nicht zufällig ist mit einem Vertrauen
von 99%.
Für die Beispiele in diesem Dokument, ein
wurde gewählt, um 0,01. Beachten Sie, dass in vielen Fällen die Parameter in den Beispielen nicht zu den empfohlenen Werten entsprechen, die Beispiele werden zur Veranschaulichung
Zwecken.

1-4

Analyse

1.1.6
Überlegungen zur Zufälligkeit, Unvorhersehbarkeit und-prüfung
Die folgenden Annahmen werden in Bezug auf Random Binary-Sequenzen zu prüfenden gemacht:

1.
Gleichmäßigkeit: An jedem Punkt in der Erzeugung einer Folge von Zufallszahlen oder Pseudo-
Bits, ist das Auftreten einer Null oder Eins gleich wahrscheinlich, dh die Wahrscheinlichkeit eines jeden
genau 1 / 2. Die erwartete Anzahl von Nullen (oder eine) ist n / 2, wobei n = die Reihenfolge
Länge.
2.
Skalierbarkeit: Jeder Test für eine Sequenz kann auch auf Teilfolgen angewendet werden
extrahiert nach dem Zufallsprinzip. Ist eine Folge zufälliger, dann eine solche Teilfolge extrahiert
sollte auch zufällig sein. Daher sollte jeder extrahiert Teilfolge jeder Test für pass Zufälligkeit.
3.
Konsistenz: Das Verhalten eines Generators muss über Startwerte konsequente (Samen). Es ist nicht ausreichend, um ein PRNG auf den Ausgang aus einem einzigen Samen-basierter Test, oder ein
RNG auf der Grundlage einer Leistung von einem einzigen physikalischen Ausgang erzeugt.
1.2 Begriffsbestimmungen
Begriff Definition
Asymptotische Analysis ein statistisches Verfahren, dass die Beschränkung der

Näherungen ergibt sich für Funktionen
von Interesse.
Asymptotische Verteilung Die Begrenzung der Verteilung einer Teststatistik die entstehen, wenn n gegen
Unendlichkeit.
Bernoulli Zufällige
Variable
Eine Zufallsvariable, die den Wert eins annimmt mit der Wahrscheinlichkeit p und die Wert von Null mit der Wahrscheinlichkeit 1-p.
Binary Sequence Eine Sequenz von Nullen und Einsen.
Binomialverteilung Eine Zufallsvariable ist binomialverteilt, wenn es eine natürliche Zahl n und eine
Wahrscheinlichkeit p, dass die Zufallsvariable ist die Anzahl der Erfolge in n unabhängige Bernoulli-Experimente, in denen die Wahrscheinlichkeit des Erfolgs in einem
einzigen Experiment ist S. In einem Bernoulli-Experiment, gibt es nur zwei möglichen Ergebnisse.
Bit String eine Folge von Bits.
Block A Teilmenge von einem Bit-String. Ein Block hat eine vorgegebene Länge.
Zentraler Grenzwertsatz Für eine Zufallsstichprobe der Größe n aus einer Population mit Mittelwert μ und
Varianz s2, ist die Verteilung der Probe bedeutet annähernd normal
mit Mittelwert μ und Varianz s2 / n als Stichprobengröße steigt.
Ergänzende Fehler
Funktion
Siehe erfc.
Konfluenten hypergeometrischen
Funktion
Die konfluente hypergeometrische Funktion ist definiert als
.
Kritischen Wert Der Wert, der durch die Teststatistik mit einer geringen Wahrscheinlichkeit überschritten wird
(Signifikanzniveau). Ein "look-up" oder berechnete Wert einer Teststatistik (dh eine Teststatistik-Wert), dass durch den Bau, hat eine kleine Wahrscheinlichkeit, auftreten (zB 5%), wenn die Null-Hypothese der Zufälligkeit ist wahr.
Verteilungsfunktion
Function (CDF) F (x)
Eine Funktion gibt die Wahrscheinlichkeit, dass die Zufallsvariable X kleiner oder ist gleich x, für jeden Wert x. Das heißt,
F (x) = P (X =
X).

Analyse

Entropie ein Maß für die Unordnung oder Zufälligkeit in einem geschlossenen System.
Die Entropie
der Unsicherheit einer Zufallsvariable X mit den Wahrscheinlichkeiten pi, ..., ist pn
definiert werden, um H (X) = werden.
Entropy Source Eine physikalische Quelle von Informationen, deren Ausgang entweder
zu sein scheint
random in selbst oder durch die Anwendung einiger Filter-/ Destillation. Dieser
Ausgang wird als Eingabe entweder eine RNG oder PRNG verwendet.
Erfc Die komplementäre Fehlerfunktion erfc (z) wird in Abschnitt 5.5.3 definiert. Dieser
Aufgabe ist es, den normalen cdf verwandt.
Geometrische Zufällige
Variable
Eine Zufallsvariable, die den Wert k annimmt, eine nicht-negative ganze Zahl mit
Wahrscheinlichkeit pk (1-p). Die Zufallsvariable x ist die Anzahl der Erfolge
bevor ein Fehler in einer unendlichen Reihe von Bernoulli-Versuchen.
Globale Struktur / Global
Wert
Eine Struktur / Wert, der zur Verfügung von allen Routinen in den Test-Code ist.
igamc Die unvollständige Gamma-Funktion Q (a, x) ist in Abschnitt 5.5.3 definiert.
Unvollständige Gamma
Funktion
Siehe die Definition für igamc.
Hypothese (Alternative) eine Erklärung Ha, dass ein Analyst als wahr (zB Ha halten
wird: die Reihenfolge
ist nicht zufällig), ob und wann die Nullhypothese bestimmt falsch zu sein.
Hypothese (Null) eine Erklärung über die H0 angenommen Default-Zustand /
Eigenschaft der
beobachteten Sequenz. Für die Zwecke dieses Dokuments, die Nullhypothese
H0 ist, daß die Reihenfolge zufällig ist. Wenn H0 ist in der Tat wahr ist, dann die
Referenz
Vertrieb und kritischen Werte der Teststatistik abgeleitet werden kann.
Kolmogorov-Smirnov-Test Ein statistischer Test, mit dem ermittelt werden kann, wenn
eine Reihe von Daten stammen aus
einer bestimmten Wahrscheinlichkeitsverteilung.
Signifikanzniveau (a) Die Wahrscheinlichkeit, dass fälschlicherweise die Ablehnung der
Nullhypothese, dh die Wahrscheinlichkeit,
der Schlussfolgerung, dass die Nullhypothese falsch ist, wenn die Hypothese ist, in
Tatsächlich wahr. Der Tester wählt in der Regel diesen Wert, typische Werte sind 0,05,
0,01 oder 0,001; gelegentlich sind kleinere Werte wie wie 0,0001 verwendet. Die
Signifikanzniveau ist die Wahrscheinlichkeit Schluss, dass eine Sequenz
nicht zufällig, wenn es in der Tat ist zufällig. Synonyme: Typ-I-Fehler, eine
Fehler.
Linear Dependence Im Rahmen der binären Matrix Rang-Test bezieht sich lineare
Abhängigkeit von m-
Bit-Vektoren, die als Linearkombination der linear ausgedrückt werden kann
unabhängige m-Bit-Vektoren.
Maple Eine interaktive Computer-Algebra-System, das eine komplette bietet
mathematische Umgebung für die Manipulation und Vereinfachung der

symbolische algebraische Ausdrücke, beliebige erweiterte Genauigkeit der Mathematik, zwei-und dreidimensionale Grafiken und Programmierung.
MATLAB Eine integrierte, technische Computer-Umgebung, die numerische vereint Berechnung, moderne Grafik und Visualisierung, sowie ein hohes Maß Programmiersprache. MATLAB enthält Funktionen für die Datenanalyse und Visualisierung; numerische und symbolische Berechnungen, technische und wissenschaftliche Grafiken, Modellierung, Simulation und Prototyping, und Programmierung, Anwendungs-Entwicklung und ein GUI-Design.
Normal (Gauss)
Verteilung
Eine kontinuierliche Verteilung, deren Dichte ist gegeben durch
, Wo μ
und s
sind die Lage und Größe
Parameter.

Analyse

P-Wert die Wahrscheinlichkeit (unter der Nullhypothese des Zufalls), dass die gewählte Teststatistik übernimmt Werte, die gleich oder schlechter sind als die beobachtet Teststatistik Wert, wenn man die Null-Hypothese. Die pValue wird oft als der "Schwanz Wahrscheinlichkeit."
Poisson Distribution - § 3.8 Poisson-Verteilungen Modell (einige) diskrete Zufallsvariablen. Typischerweise
einer Poisson-Zufallsvariable ist eine Zählung der Anzahl der seltenen Ereignisse, die treten in einem bestimmten Zeitintervall.
Wahrscheinlichkeitsdichte
Function (PDF)
Eine Funktion, die "local" Wahrscheinlichkeitsverteilung einer Test liefert Statistik. Von einer endlichen Stichprobenumfang n wird eine Wahrscheinlichkeitsdichtefunktion werden
näherungsweise durch ein Histogramm.
Wahrscheinlichkeitsverteilung die Zuordnung einer Wahrscheinlichkeit, die möglichen Ergebnisse (Realisierungen) von
eine Zufallsvariable.
Pseudo-Zufallszahlen
Generator (PRNG)
Eine deterministische Algorithmus, der eine wirklich zufällige binäre Folge von bestimmten
Länge k, eine binäre Folge der Länge l>> k was zu sein scheint
zufällig. Der Eingang zum Generator wird als Samen, während der Ausgang ist genannt Pseudo-Bit-Sequenz.
Random Number
Generator (RNG)

Ein Mechanismus, der wirklich zufällige Daten generiert vorgibt.
Random Binary Sequence Eine Sequenz von Bits, für die die Wahrscheinlichkeit jedes Bit eine "0" oder "1"
ist ½. Der Wert jedes Bit ist unabhängig von allen anderen etwas in der Reihenfolge, d.h. wird jedes Bit unberechenbar.
Zufallsvariable Zufalls-Variablen unterscheiden sich von den üblichen deterministischen Variablen (der Wissenschaft
und Technik) in die Zufallsvariablen ermöglichen die systematische
Verteilungswirkungen Zuordnung der Wahrscheinlichkeitswerte zu jedem möglichen Ergebnis.
Rank (einer Matrix) Bezieht sich auf den Rang einer Matrix in der linearen Algebra über GF (2). Nachdem
reduziert eine Matrix in zeilengestaffelte Form über elementare Zeilenoperationen, die viele von Null verschiedene Reihen, wenn überhaupt, werden gezählt, um die Bestimmung
Anzahl der linear unabhängigen Zeilen oder Spalten in der Matrix.
Führen Sie eine ununterbrochene Folge von Bits, wie (dh entweder nur Nullen oder Einsen).
Seed Der Eingang zu einem Pseudo-Zufallszahlen-Generator. Verschiedene Samen erzeugen
verschiedenen Pseudo-Sequenzen.
SHA-1 Der Secure Hash Algorithm in Federal Information Processing definiert Standard 180-1.
Standard-Normal-
Verteilungsfunktion
Funktion
Siehe die Definition in Abschnitt 5.5.3. Dies ist die normale Funktion für Mittel =
0 und Varianz = 1 ist.
Statistisch unabhängig
Events
Zwei Ereignisse sind unabhängig, wenn das Auftreten eines Ereignisses nicht beeinflusst
die Chancen für das Auftreten der anderen Veranstaltung. Das mathematische Formulierung der Unabhängigkeit der Ereignisse A und B ist die Wahrscheinlichkeit des Vorkommen von sowohl A als auch B gleich dem Produkt der Wahrscheinlichkeiten von A und B (d. h. P (A und B) = P (A) P (B)).
Statistische Test (einer
Hypothesis)
Eine Funktion der Daten (Binary Stream), die berechnet und wird verwendet, um entscheiden, ob die Nullhypothese abzulehnen. Eine systematische statistische Regel, deren Zweck es ist, eine Aussage über zu generieren, ob die Experimentator sollte annehmen oder ablehnen die Nullhypothese Ho.
Word-Eine vordefinierte Zeichenkette, bestehend aus einem festen Muster / template (zB 010,
0110).

Analyse

1.3 Abkürzungen
Abkürzung Definition
ANSI American National Standards Institute
FIPS Federal Information Processing Standard
NIST National Institute of Standards and Technology
Random Number Generator RNG
SHA-1 Secure Hash Algorithm

1,4 mathematische Symbole
In der Regel wird die folgende Notation in diesem Dokument verwendet. Doch die Tests in diesem Dokument
wurden entwickelt und beschrieben von mehreren Autoren, die etwas andere Schreibweise verwendet haben.
Der Leser wird empfohlen, die Schreibweise für jeden Test getrennt von Notation in anderen verwendet betrachten
Tests.

Symbol Bedeutung
. X.
Der Boden in Abhängigkeit von x, für eine gegebene reelle positive x, x..
= X-g, wo. X.
ist eine nicht-negative ganze Zahl ist, und 0 =
g <1 ist.
ein
Das Signifikanzniveau.
d Die normalisierte Differenz zwischen den beobachteten und erwarteten Anzahl der Frequenz-
Komponenten. Siehe Abschnitte 2.6 und 3.6.
.. 2
m (obs);
.
2,2
m (obs)
Ein Maß dafür, wie auch die beobachteten Werte mit den erwarteten Wert. Siehe Abschnitte
2.11 und 3.11.
E [] Der Erwartungswert einer Zufallsvariable.
e
Der ursprüngliche Eingang Kette von Null und Eins-Bits getestet werden.
ei der i-te Bit in der ursprünglichen Reihenfolge e.
H0 Die Nullhypothese, dh die Aussage, dass die Reihenfolge zufällig ist.
log (x) Der natürliche Logarithmus von x: log (x) = loge (x) = ln (x).
log2 (x)
Definiert als, wobei ln der natürliche Logarithmus.

M Die Anzahl der Bits in einem substring (Block) getestet.
N die Anzahl der M-Bit-Blöcke zu testen.
n Die Anzahl der Bits in den Bach in der Testphase.
fn Die Summe der Abstände zwischen log2 passenden L-Bit-Vorlagen, dh die Summe der
Anzahl der Stellen in den Abstand zwischen L-Bit-Vorlagen. Siehe Abschnitte 2.9 und 3.9.
p
3,14159 ..., sofern nicht anders für einen bestimmten Test definiert.
s
Die Standardabweichung einer Zufallsvariablen =.
s2 Die Varianz einer Zufallsvariable = (Standardabweichung) 2.
Schluchzen Der beobachtete Wert, die als eine Statistik in der Frequenz-Test verwendet wird.
Sn Die n-te Partialsumme für Werte Xi = {-1, +1}, dh die Summe der ersten n Werte von Xi.
S
Die Summation Symbol.
F
Standard-Normal-Verteilungsfunktion (siehe Abschnitt 5.5.3).
. J Die Gesamtzahl der Zeiten, die einem bestimmten Zustand in den identifizierten Zyklen auftritt. Sehen
Abschnitt 2.15 und 3.15.
Xi Die Elemente der Zeichenfolge, die aus ± 1, die auf Zufälligkeit getestet werden soll, wo
Xi = 2ei -1.

Analyse

.2 Die [theoretische] Chi-Quadrat-Verteilung, als Teststatistik verwendet, auch eine Teststatistik
dass folgt der 0,2-Distribution.
.2 (Obs) Der Chi-Quadrat-Statistik auf den beobachteten Werten berechnet. Siehe Abschnitte 2.2, 2.4, 2.5,
2,7, 2,8, 2,10, 2,12, 2,14, und die entsprechenden Abschnitte des § 3.
Vn Die erwartete Anzahl von Runs, die in einer Folge der Länge n unter einer auftreten würden
Annahme des Zufalls siehe Abschnitte 2.3 und 3.3.
Vn (obs) Die beobachtete Anzahl der Durchläufe in einer Folge der Länge n. Siehe Abschnitte 2.3 und 3.3.

Analyse

2. Random Number Generation Tests
Das NIST-Test-Suite ist ein statistisches Paket, bestehend aus 15 Tests, die entwickelt, um den Test wurden
Zufälligkeit (beliebig lange) Binärsequenzen entweder Hardware oder Software produziert
kryptographische zufällige oder Pseudo-Zufallszahlen-Generatoren. Diese Tests auf einer Vielzahl von unterschiedlichen Schwerpunkten
Typen von Nicht-Zufälligkeit, dass in einer Sequenz existieren könnte. Einige Tests sind in einer Vielzahl von abbaubaren
Untertests. Die 15 Tests sind:

1. Die Frequenz (Monobit) Test,
2. Frequency-Test innerhalb eines Blocks,
3. Die Testläufe,
4. Tests für die Längste-Run-of-One in einem Block,
5. Die binäre Matrix Rank-Test,
6. Die Diskrete Fourier-Transformation (Spectral) Test,
7. Die Non-überlappenden Template Matching Test,
8. Die Überlappung Template Matching Test,
9. Maurer ist "Universal Statistical" Test,
10. Die lineare Komplexität Test,
11. Die Serial-Test,
12. Das Ungefähre Entropy-Test,
13. Die Summenwerte (Cusums) Test,
14. The Random Excursions Test, und
15. The Random Excursions Variant Test.

Dieser Abschnitt (Abschnitt 2) besteht aus 15 Abschnitten, einem Teilabschnitt für jeden Test. Jeder Unterabschnitt enthält
ein hohes Niveau Beschreibung des jeweiligen Test. Die entsprechenden Abschnitte in Abschnitt 3 stellen die
technische Details für jeden Test.

Abschnitt 4 enthält eine Erörterung der Prüfstrategie und die Interpretation der Testergebnisse. Die Reihenfolge der
Anwendung der Tests im Test-Suite ist beliebig. Es ist jedoch empfohlen, dass die Frequenz-Test
werden zum ersten Mal ausgeführt, da dies liefert die grundlegende Beweis für die Existenz von Nicht-Zufälligkeit in einem
Sequenz, insbesondere Nicht-Uniformität. Wenn dieser Test fehlschlägt, ist die Wahrscheinlichkeit, dass andere Tests fehlschlagen hoch.
(Hinweis: Die meiste Zeit in Anspruch statistischer Test ist die lineare Komplexität Test, siehe § § 2,10 und
3,10).

Abschnitt 5 bietet eine Bedienungsanleitung für die Einrichtung und Durchführung der

Tests und eine Diskussion über Programm-Layout.
Das Statistik-Paket beinhaltet den Quellcode und Sample-Daten-Sets. Der Test-Code wurde in ANSI entwickelt

C. Einige Eingänge sind davon ausgegangen, dass die globalen Werte als Aufrufparameter werden.
Eine Reihe von Tests in die Test-Suite haben die Standardnormalverteilung und der Chi-Quadrat (! 2) als Referenz
Distributionen. Wenn die Sequenz unter Test ist in der Tat nicht zufällig, werden die berechneten Testgröße im Herbst
extreme Regionen der Referenz-Distribution. Die Standard-Normalverteilung (dh die glockenförmigen
Kurve) wird verwendet, um den Wert der Teststatistik von der RNG mit dem erwarteten Wert der erhaltenen vergleichen

der Statistik unter der Annahme der Zufälligkeit. Die Teststatistik für die Standard-Normalverteilung ist
der Form z = (x-μ) / s, wobei x der Probe Teststatistik-Wert und μ und s2 sind die zu erwartenden Wert
und die Varianz der Teststatistik. Die! 2distribution (dh, eine linke schiefe Kurve) wird verwendet, um den Vergleich
der beobachteten Frequenzen einer Probe messen die Güte der Anpassung an die entsprechenden erwarteten

Analyse

Frequenzen der Hypothese Verteilung. Die Teststatistik ist von der Form

! 22 = wo oi und ei den beobachteten und erwarteten Häufigkeiten der Maßnahme bzw. sind.

Für viele der Tests in dieser Test-Suite hat die Annahme getroffen, dass die Größe der Sequenzlänge,
n, ist groß (in der Größenordnung von 103 bis 107). Für solche große Stichproben von n, asymptotische Referenzverteilungen
abgeleitet wurden und auf die Durchführung der Tests. Die meisten Tests sind für kleinere Werte von

n. Wenn jedoch für kleinere Werte von n verwendet wird, würde das asymptotische Referenzverteilungen unangemessen
und müsste durch eine genaue Distributionen, die häufig nur schwer zu berechnen, ersetzt werden würde.
Hinweis: Für viele der Beispiele in Abschnitt 2, sind kleine Probenmengen zur Veranschaulichung verwendet

lediglich, z. B., n = 10. Die normale Näherung ist nicht wirklich geeignet für diese Beispiele.

2.1 Häufigkeit (Monobit) Test
2.1.1
Testzwecken
Der Schwerpunkt des Tests liegt der Anteil von Nullen und Einsen für die gesamte Sequenz. Der Zweck dieses Tests
ist, um festzustellen, ob die Anzahl der Einsen und Nullen in einer Folge etwa sind die gleichen wie würde
für eine wirklich zufällige Sequenz zu erwarten. Der Test prüft die Nähe des Bruchs von Einsen zu ½, dass
ist, sollte die Anzahl von Einsen und Nullen in einer Serie über die gleichen sein. Alle nachfolgenden Tests abhängen
Im Laufe der diesen Test.

2.1.2
Function Call
Frequency (n), wobei:

n
Die Länge des Bit-String.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.1.3
Teststatistik und Reference Vertrieb
Schluchzen:
Der absolute Wert der Summe der Xi (wo, Xi = 2e
-1 = ± 1) in der Reihenfolge verteilt
durch die Quadratwurzel aus der Länge der Sequenz.

Die Referenz-Verteilung für die Teststatistik ist die Hälfte normal (für große n). (Hinweis: Wenn z (wo

, Siehe Abschnitt 3.1) als normal verteilt ist, dann | z | ist die Hälfte normalverteilt) Wenn die.

Reihenfolge ist zufällig, dann die Plus-und Minus bringen wird eher gegenseitig aufheben, so dass der Test
Statistik wird über 0 sein. Wenn es zu viele Einsen oder zu viele Nullen sind, dann ist die Teststatistik wird tendenziell
größer als Null ist.

2.1.4
Test Beschreibung
(1)
Die Umstellung auf ± 1: Die Nullen und Einsen der Eingabesequenz (e) sind auf Werte von -1 und umgesetzt
+1 Und sind gemeinsam zu produzieren hinzugefügt
SXXXnn = + + + wo Xi = 2ei - 1.

Analyse

Zum Beispiel, wenn e
= 1011010101, dann n = 10 und Sn = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + (-1) + 1 = 2.

(2) Berechnen Sie die Prüfgröße Schluchzen =
Snn
Für das Beispiel in diesem Abschnitt, schluchzt =
102 = 0,632455532.

(3) Berechnen Sie P-Wert = erfc
$ $%
&
2sobs, wo erfc ist die komplementäre Fehlerfunktion im Sinne
Abschnitt 5.5.3.

Für das Beispiel in diesem Abschnitt, P-Wert = erfc $%
&

2.1.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.1.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 3 des Abschnitts 2.1.4 erhält man =
0,01 (dh, P-Wert = 0,527089), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass, wenn der P-Wert waren klein (<0,01), dann ist dies durch verursacht werden würde

Sn

oder

OBSS
sind groß.

Große positive Werte von Sn sind bezeichnend für zu viele Lieben, und große negative Werte von Sn sind indikativ
zu viele Nullen.

2.1.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 100 Bit bestehen (dh, n =
100).

2.1.8 Beispiel
(Eingang) e
= 11001001000011111101101010100010001000010110100011
00001000110100110001001100011001100010100010111000
(Eingang) n = 100
(Verarbeitung) S100 = -16
(Verarbeitung) Schluchzen = 1,6
(Output) P-Wert = 0,109599
(Schluss) Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2-3

Analyse

2,2-Frequenz-Test innerhalb eines Blocks
2.2.1 Test-Purpose
Der Schwerpunkt des Tests liegt der Anteil der Einsen in M-Bit-Blöcke. Der Zweck dieser Prüfung soll festgestellt werden
ob die Häufigkeit der Einsen in einem M-Bit-Block ist etwa M / 2, wie unter einer zu erwarten wäre
Annahme des Zufalls. Für Blockgröße M = 1, verkommt dieser Test 1 Test, der Frequenz (Monobit)
zu testen.

2.2.2 Function Call
BlockFrequency (M, n), wobei:
M Die Länge der einzelnen Blöcke.
n Die Länge des Bit-String.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:
e

Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.2.3 Teststatistik und Reference Vertrieb
2!
(Obs): Ein Maß dafür, wie gut die beobachteten Anteil der Einsen in einem bestimmten M-Bit
Block mit dem erwarteten Anteil (2.1).

Die Referenz-Verteilung für die Teststatistik ist ein 0,2-Distribution.

2.2.4 Test Beschreibung
(1) Partition der Eingabesequenz in N =
! "!
$
#
Mn
nicht überlappende Blöcke. Entsorgen Sie alle unbenutzten Bits.

Zum Beispiel, wenn n = 10, M = 3 und e
= 0110011010, 3 Blocks (N = 3) geschaffen würden,
bestehend aus 011, 001 und 101. Die endgültige 0 würde verworfen werden.

(2) Bestimmen Sie den Anteil pi von Einsen in jeder M-Bit-Block mit der Gleichung
für 1 =
i =
N.
Für das Beispiel in diesem Abschnitt, p1 = 2 / 3, p2 = 1 / 3 und p3 = 2 / 3.

(3) Berechnen Sie die 0,2-Statistik: .2 (obs) = 4M
(
iN =
!
1PI-½) 2.
ich
=
$
,
Für das Beispiel in diesem Abschnitt, .2 (obs) = 4 x3 x

()()()$%
&.
2-4

Analyse

(4)
Berechnen Sie P-Wert = igamc (N / 2, .2 (obs) / 2), wo igamc ist die unvollständige Gamma-Funktion für
Q (a, x) wie in Abschnitt 5.5.3 definiert.
Hinweis: Beim Vergleich dieser Abschnitt gegen die technische Beschreibung in Abschnitt 3.2 zu beachten, dass
Q (a, x) = 1-P (a, x).

Für das Beispiel in diesem Abschnitt, P-Wert = igamc $%
&

2.2.5
Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.2.6
Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 4 des Abschnitts 2.2.4 erhält man =
0,01 (dh, P-Wert = 0,801252), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass kleine P-Werte (<0,01) wäre eine große Abweichung von der gleichen Anteil angegeben haben
Einsen und Nullen in mindestens einem der Blöcke.

2.2.7
Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 100 Bit bestehen (dh, n =
100). Note
dass n =
MN. Die Blockgröße M sollte so gewählt sein, dass M =
20, M> 0,01 n und N <100.

2.2.8
Beispiel
(Eingang) e
= 11001001000011111101101010100010001000010110100011
00001000110100110001001100011001100010100010111000
(Eingang) n = 100
(Eingang) M = 10
(Verarbeitung) N = 10
(Verarbeitung) .2 = 7.2

(Output) P-Wert = 0,706438
(Schluss) Da P-Wert =
0,0, akzeptieren die Sequenz als zufällig.

2,3 Runs-Test
2.3.1
Testzwecken
Der Fokus dieses Tests ist die Gesamtzahl der Durchläufe in der Sequenz, wo ein Lauf ist eine ununterbrochene Folge
von identischen Bits. Ein Run der Länge k besteht aus genau k identisch Bits und wird vor und nach mit beschränkter
ein bisschen von der entgegengesetzten Wert. Der Zweck der Testläufe ist es, festzustellen, ob die Anzahl der Pisten

Analyse

Einsen und Nullen in verschiedenen Längen als ein zufälliger Reihenfolge erwartet. Insbesondere wird getestet,
ob die Oszillation zwischen diesen Nullen und Einsen zu schnell oder zu langsam.

2.3.2
Function Call
Runs (n), wobei:
n Die Länge des Bit-String.
Zusätzliche Eingänge für die Funktion, sondern versorgt die Testing-Code:
e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.3.3
Teststatistik und Reference Vertrieb
Vn (obs):
Die Gesamtzahl der läuft (dh, die Gesamtzahl der Null läuft + die Gesamtzahl der one-Läufe) über alle n Bits.

Die Referenz-Verteilung für die Teststatistik ist ein 0,2-Distribution.

2.3.4
Test Beschreibung
Hinweis: Die Testläufe führt eine Frequenz-Test als Voraussetzung.

(1)
Berechnen Sie die Pre-Test Anteil p

der Einsen in der Eingangssequenz:
Zum Beispiel, wenn e
= 1001101011, dann n = 10 und p
= 10.06 = 05.03.

(2)
Bestimmen Sie, ob die Voraussetzung Frequency Test ist bestanden: Wenn
nachgewiesen werden kann, dass ", dann die angezeigt werden
Läuft Test muss nicht durchgeführt werden (dh, sollte der Test nicht, weil eine
Verweigerung der überfahren worden
Pass-Test 1, die Frequenz (Monobit)-Test). Wenn der Test nicht anwendbar ist, dann ist
die
! P-Wert wird eingestellt
0.0000. Beachten Sie, dass für diesen Test

wurde in den Test-Code vorgegeben.

Für das Beispiel in diesem Abschnitt, da
= "
, Dann | p
-1 / 2 | = | 3 / 5 - 1 / 2 | = 0,1 <t,
und der Test wird nicht ausgeführt.
Da der beobachtete Wert p
ist innerhalb der ausgewählten Rahmen wird die Testläufe anwendbar.

(3) Berechnen Sie die Prüfgröße
obs (V1n1kn + =, wobei r (k) = 0, wenn ek = ek +1, und r (k) = 1 sonst.
Da e
= 1 00 11 0 1 0 11, dann

V10 (obs) = (1 +0 +1 +0 +1 +1 +1 +1 +0) +1 = 7.

(4)
Berechnen Sie P-Wert = erfc
=
#.
$ $%
&
2) (|
2-6

Analyse

1. Fr das Beispiel P-Wert = erfc = 0,147232.

$ $%
• • • •
2.3.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.3.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 4 des Abschnitts 2.3.4 erhält man =
0,01 (dh, P-Wert = 0,147232), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass ein großer Wert für Vn (obs) würde eine Schwingung in der Zeichenfolge, die zu schnell ist angegeben haben, ein
kleinen Wert wäre gezeigt, dass die Schwingung zu langsam sind. (Eine Schwingung wird als sein
Wechsel von einer auf eine Null oder umgekehrt.) Eine schnelle Oszillation tritt auf, wenn es eine Menge von Änderungen, zB
010101010 schwingt mit jedem Bit. Ein Bach mit einer langsamen Schwingung hat weniger läuft als wäre
erwartet in einer zufälligen Reihenfolge, zB eine Sequenz mit 100 diejenigen, um 73 Nullen, gefolgt von
127 Einsen (insgesamt 300 Bit) hätte nur drei Runs, während 150 läuft zu erwarten wäre.

2.3.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 100 Bit bestehen

(dh, n =
100).

2.3.8 beispiel
(Eingang) e
= 11001001000011111101101010100010001000010110100011
00001000110100110001001100011001100010100010111000
(Eingang) n = 100
(Eingang) t
= 0,02
(Verarbeitungs-) p
= 0,42
(Verarbeitung) Vn (obs) = 52
(Output) P-Wert = 0,500798
(Schluss) Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2.4 Prüfung für den Longest Run of Ones in einem Block
2.4.1 Test-Purpose
Der Schwerpunkt der Prüfung ist die längste Abfahrt der Einsen in M-Bit-Blöcke. Der Zweck dieses Tests ist es,
festzustellen, ob die Länge der längsten Piste der Einsen innerhalb der getesteten Sequenz im Einklang mit den sich
Länge der längsten Piste der diejenigen, die in zufälliger Reihenfolge zu erwarten wäre. Beachten Sie, dass eine Unregelmäßigkeit in
der erwarteten Länge des längsten Lauf von Einsen impliziert, dass es auch eine Unregelmäßigkeit in der erwarteten
Länge der längsten Piste von Nullen. Daher ist nur ein Test für diejenigen notwendig. Siehe auch Abschnitt 4.4.

Analyse

2.4.2
Function Call
LongestRunOfOnes (n), wobei:
n Die Länge des Bit-String.

Zusätzlicher Eingang für die Funktion durch die Erprobung Code geliefert:

Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als eine globale Struktur zu

die Zeit der Aufruf der Funktion, e
= E1, E2, ..., en.

M Die Länge der einzelnen Blöcke. Der Test-Code wurde so voreingestellt, dass drei Werte für Aufnahme

M: M = 8, M = 128 und M = 104 in Übereinstimmung mit den folgenden Werten der Sequenz
Länge n:
Mindestens n M
128 8
6272 128
750,000 104

N
Die Anzahl der Blöcke, ausgewählte in Übereinstimmung mit dem Wert des M.

2.4.3
Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachteten längste Abfahrt Länge im M-Bit-Blöcke entspricht dem erwarteten längste Länge in M-Bit-Blöcke.

Die Referenz-Verteilung für die Teststatistik ist ein 0,2-Distribution.

2.4.4
Test Beschreibung
(1)
Teilen Sie die Sequenz in M-Bit-Blöcke.
(2)
Tabulate die Frequenzen. I der längsten Abfahrten der Einsen in jedem Block in Kategorien, wobei jede
Zelle enthält die Anzahl der Durchläufe von Einsen einer bestimmten Länge.
Für die Werte von M von der Test-Code unterstützt wird, wird die vi-Zellen halten die folgenden zählen:

vi M = 8 M = 128 M = 104
v0 =
1 =
4 =
10
v1 2 5 11
v2 3 6 12
v3 =
4 7 13
v4 8 14
v5 =
9 15
v6 =
16

Ein Beispiel finden Sie in Abschnitt 2.4.8.

Analyse

(3) Berechnen Sie, wo Sie die Werte für pi in Abschnitt 3.4 zur Verfügung. Sterben
obs (
$
Werte von K und N sind durch den Wert von M in Übereinstimmung mit der folgenden Tabelle bestimmt:

M	K	N
8	3	16
128	5	49
104	6	75

Für das Beispiel 2.4.8,

"
(4) Berechnen Sie P-Wert = igamc
$ $%
&
.
Für das Beispiel P-Wert = igamc 0,180598.

2.4.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.4.6 Fazit und Interpretation der Ergebnisse
Für das Beispiel in Abschnitt 2.4.8, da die P-Wert =
0,01 (p-Wert = 0,180609), ist die Schlussfolgerung, dass die
Reihenfolge ist zufällig. Beachten Sie, dass große Werte von .2 (obs) zeigen, dass die getesteten Sequenz-Cluster hat
diejenigen.

2.4.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens Bits besteht, wie in der Tabelle angegebenen
in Abschnitt 2.4.2.

2.4.8 beispiel
Für den Fall, K = 3 und M = 8:

(Eingang) e

= 11001100000101010110110001001100111000000000001001
00110101010001000100111101011010000000110101111100
11001110011011011000010110010

(Eingang) n = 128
(Verarbeitung) Nebensatz M11001100
01101100
11100000
ax-Run
(2)
(2)
(3)
Nebensatz M00010101
01001100
00000010
ax-Run
(1)
(2)
(1)

2-9

Analyse

01001101 (2) 01010001 (1)
00010011 (2) 11010110 (2)
10000000 (1) 11010111 (3)
11001100 (2) 11100110 (3)
11011000 (2) 10110010 (2)

(Verarbeitung) .0 = 4; .1 = 9; 2 = 3; .4 = 0; .2 = 4,882457
(Output) P-Wert = 0,180609
(Schluss) Da die P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,5 Binary Matrix Rank Test
2.5.1 Test-Purpose
Der Schwerpunkt der Prüfung ist der Rang disjunkte Untermatrizen der gesamten Sequenz. Der Zweck dieses Tests ist
um zu überprüfen, für die lineare Abhängigkeit zwischen fester Länge Teilstrings der ursprünglichen Sequenz. Beachten Sie, dass dieser Test
erscheint auch in der DIEHARD Reihe von Tests [7].

2.5.2 Function Call
Rank (n), wobei:

n Die Länge des Bit-String.
Zusätzlicher Eingang von der Funktion durch den Test-Code geliefert verwendet:
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als eine globale Struktur zu

die Zeit der Aufruf der Funktion, e
= E1, E2, ..., en.
M Die Anzahl der Zeilen in jeder Matrix. Für die Test-Suite, hat M auf 32 gesetzt wurde. Wenn andere
Werte von M verwendet werden, müssen neue Annäherungen an berechnet werden.
Q Die Anzahl der Spalten in jeder Matrix. Für die Test-Suite hat Q auf 32 gesetzt wurde. Wenn andere
Werte von Q verwendet werden, müssen neue Annäherungen an berechnet werden.

2.5.3 Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachtete Anzahl der Reihen der verschiedenen Ordnungen entsprechen den
erwartete Anzahl von Reihen unter der Annahme der Zufälligkeit.

Die Referenz-Verteilung für die Teststatistik ist ein 0,2-Distribution.

2.5.4 Test Beschreibung
(1) nacheinander teilen die Sequenz in M • Q-Bit disjunkte Blöcke, es wird bestehen Blöcke. Ausrangierte Bits wird berichtet, als nicht in die Berechnung in jedem Block verwendet werden.
Sammeln Sie die M • Q-Bit-Segmente in M von Q-Matrizen. Jede Zeile der Matrix ist gefüllt mit
aufeinanderfolgenden Q-Bit-Blöcke der ursprünglichen Sequenz e.

=
2-10

Analyse

Zum Beispiel, wenn n = 20, M = Q = 3 und e
= 01011001001010101101 und partitionieren dann den Strom

in N =
$
#
• 33n = 2 Matrizen der Kardinalität M • Q (3 • 3 = 9). Beachten Sie, dass die letzten beiden Bits (0 und 1)
werden verworfen. Die beiden Matrizen sind und
110101010. Beachten Sie, dass die erste Matrix

010011010
besteht aus den ersten drei Bits in Zeile 1, die zweite Gruppe von drei Bits in Zeile 2, und die dritte Gruppe von
drei Bits in Zeile 3. Die zweite Matrix ist ähnlich konstruiert mit der nächsten neun Bits in der
Sequenz.

(2)
Bestimmen Sie die Binär-Rang (
lR) jeder Matrix, wo
1 = l. Die Methode zur Bestimmung
der Rang in Anhang A beschrieben
Für das Beispiel in diesem Abschnitt ist der Rang der ersten Matrix 2 (R1 = 2) und den Rang der
zweite Matrix 3 (R2 = 3).

(3) Lassen FM = die Anzahl der Matrizen mit
lR
lR = M (vollen Rang),
FM-1 = Anzahl der Matrizen mit = M-1 (vollen Rang -1),
N-FM-FM-1 = Anzahl der Matrizen übrig.

Für das Beispiel in diesem Abschnitt, FM = F3 = 1 (R2 hat die volle Rang 3), FM-1 = F2 = 1 (R1 hat Rang
2), und keine Matrix hat keine niedrigeren Rang.

(4) Berechnen Sie
".
Für das Beispiel in diesem Abschnitt

"
= 0,596953.
(5) Compute. Da gibt es 3 Klassen im Beispiel die P-Wert für die
.
2obsevalueP "beispielsweise ist gleich
Für das Beispiel in diesem Abschnitt, P-Wert =

0,741948.

2.5.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

Analyse

2.5.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 5 des Abschnitts 2.5.4 erhält man =
0,01 (p-Wert = 0,741948), ist die Schlussfolgerung, dass
die Reihenfolge ist zufällig.

Beachten Sie, dass große Werte von

obs (2!
(Und damit kleine P-Werte) wäre eine Abweichung von der angegeben haben,

Rang Verteilung von diesem entsprechend einer zufälligen Reihenfolge.

2.5.7 Input Size Empfehlung
Die Wahrscheinlichkeiten für M = Q = 32 berechnet wurden und in den Test-Code.
Andere Möglichkeiten der
M und Q kann ausgewählt werden, aber die Wahrscheinlichkeiten müssten berechnet werden. Die minimale Anzahl von
Bits getestet werden müssen, so dass n = werden
38MQ (d. h. mindestens 38 Matrizen erstellt werden). Für M = Q = 32, die jeweils Reihenfolge geprüft werden sollte ein Minimum von 38.912 Bits bestehen.

2.5.8 beispiel
(Eingang) e
= Die ersten 100.000 Binärziffern in den Ausbau der e
(Eingang) n = 100000, M = Q = 32 (Hinweis:. 672 bit wurden verworfen)
(Verarbeitung) N = 97
(Verarbeitung) FM = 23, FM-1 = 60, N - FM-FM-1 = 14
(Verarbeitung) .2 = 1,2619656
(Output) P-Wert = 0,532069
(Schluss) Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,6 Diskrete Fourier-Transformation (Spectral) Test
2.6.1 Test-Purpose
Der Fokus dieses Tests ist die Peakhöhen in der diskreten Fourier-Transform der Sequenz. Der Zweck
dieses Tests ist es, periodische Eigenschaften (dh, sich wiederholende Muster, die einander nahe sind) in den getesteten erkennen
Sequenz, die eine Abweichung von der Annahme des Zufalls hinweisen würde. Die Absicht ist zu erkennen,
ob die Anzahl der Gipfeln über der 95%-Schwelle ist wesentlich anders als 5%.

2.6.2 Function Call
DiscreteFourierTransform (n), wobei:
n Die Länge des Bit-String.
Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

Analyse

2.6.3
Teststatistik und Reference Vertrieb
d:
Die normalisierte Differenz zwischen den beobachteten und der erwarteten Anzahl von Frequenzen
Komponenten, die über die 95% Schwelle.
Die Referenz-Verteilung für die Teststatistik ist die Normalverteilung.

2.6.4 Test Beschreibung
(1)
Die Nullen und Einsen der Eingabesequenz (e) sind auf Werte von -1 und +1 umgewandelt, um die zu schaffen
Folge X = x1, x2, ..., xn, wo xi = 2ei - 1.
Zum Beispiel, wenn n = 10 und e
= 1001010011, dann X = 1, -1, -1, 1, -1, 1, -1, -1, 1, 1.

(2)
Tragen Sie eine diskrete Fourier-Transformation (DFT) auf X zu erzeugen: S = DFT (X). Eine Folge von
komplexen Variablen erzeugt, stellt periodischen Komponenten der Folge von Bits auf verschiedenen Frequenzen (siehe Abschnitt 3.6 für eine Probe-Diagramm eines DFT Ergebnis).
(3)
Berechnen Sie M = E-Modul (S ') =
| S '|, wo S' ist der Teilstring aus den ersten n / 2 Elemente in
S, und der Modul-Funktion erzeugt eine Folge von Peakhöhen.
(4) Berechne T = = 95% Peakhöhe Schwellenwert. Unter der Annahme einer Zufälligkeit, sollten 95% der Werte aus dem Test erhaltenen nicht überschreiten T.

log10.05
"
#
$
%
&
'N
(5)

Compute N0 = .95 n / 2. N0 ist die erwartete theoretische (95%) Anzahl der Peaks (unter der
Annahme des Zufalls), die weniger als T. sind
Für das Beispiel in diesem Abschnitt, N0 = 4,75.

(6)
Compute N1 = die tatsächlich beobachteten Anzahl der Peaks in M, die weniger als T. sind
Für das Beispiel in diesem Abschnitt, N1 = 4.
(7)
Berechnen
=
.
Für das Beispiel in diesem Abschnitt
= -1,538968.

=
(8) Berechnen Sie P-Wert =
Für das Beispiel in diesem Abschnitt, P-Wert =

0,123812.

2-13

Analyse

2.6.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.6.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 8 des Abschnitts 2.6.4 erhält man =
0,01 (p-Wert = 0,123812), ist die Schlussfolgerung, dass
die Reihenfolge ist zufällig.

A d-Wert zu niedrig ist, würde zeigen, dass es zu wenige Peaks (<95%) unterhalb von T, und zu viele
Peaks (mehr als 5%) über T.

2.6.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 1000 Bit bestehen (dh, n =
1000).

2.6.8 beispiel

(Eingang) e
= 11001001000011111101101010100010001000010110100011
00001000110100110001001100011001100010100010111000
(Eingang) n = 100

(Verarbeitung) N1 = 46
(Verarbeitung) N0 = 47,5
(Verarbeitung) d = -0,973329
(Output) P-Wert = 0,330390
(Schluss) Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,7 Nicht überlappende Template Matching-Test
2.7.1 Test-Purpose
Der Fokus dieses Tests ist die Anzahl der Vorkommen von vorgegebenen Ziel-Strings. Der Zweck dieser
Tests ist es, Generatoren, dass zu viele Instanzen eines gegebenen nicht-periodische (aperiodische) Muster zu erzeugen erkennen.
Für diesen Test und für die Überlappung Template Matching Test der Abschnitt 2.8, ist ein m-Bit-Fenster verwendet werden, um
Suche nach einem bestimmten m-Bit-Muster. Wenn das Muster nicht gefunden wird, gleitet das Fenster um eine Bit-Position. Wenn der
Muster gefunden wird, wird das Fenster wieder auf das Bit nach dem gefundenen Muster, und die Suche fortgesetzt.

2.7.2 Function Call
NonOverlappingTemplateMatching (m, n)
m Die Länge in Bits jedes Template. Die Vorlage wird das Ziel-String.
n Die Länge der gesamten Bit-String unter Test.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

Analyse

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

B
Die m-Bit-Vorlage abgestimmt werden; B ist eine Kette von Einsen und Nullen (der Länge m), die
definiert in einer Template-Bibliothek von nicht-periodische Muster innerhalb der Test-Code enthalten.

M
Die Länge in Bits der Teilstring von e
getestet werden. M hat 131.072 (d. h. 217) gesetzt
in den Test-Code.

N
Die Anzahl der unabhängigen Blöcken. N hat bei 8 wurde in den Test-Code behoben.

2.7.3
Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachtete Anzahl der Vorlage "hits" entspricht der erwartete Anzahl der Vorlage "Hits" (unter der Annahme des Zufalls).

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.7.4
Test Beschreibung
(1)
Partitionieren Sie die Sequenz in N unabhängige Blöcke der Länge M.
Zum Beispiel, wenn e
= 10100100101110010110, dann n = 20. Wenn N = 2 und M = 10, dann die beiden Blöcke würden 1010010010 und 1110010110 sein.

(2)
Lassen Wj (j = 1, ..., N) die Anzahl der Male, dass B (der Vorlage) kommt innerhalb des Blocks werden j. Note
dass j = 1, ..., N. Die Suche nach Übereinstimmungen geht durch die Schaffung eines m-Bit-Fenster auf die Reihenfolge,
Vergleich der Bits innerhalb dieses Fensters gegen die Vorlage. Wenn es keine Übereinstimmung gibt, wird das Fenster
gleitet über ein Bit, wenn z. B. m = 3 und das aktuelle Fenster enthält Bits 3 bis 5, dann die nächste
Fenster enthält Bits 4 bis 6. Wenn es eine Übereinstimmung gibt, gleitet das Fenster über m Bits, zB, wenn die
Strom (erfolgreichen) Fenster enthält Bits 3 bis 5, dann im nächsten Fenster Bits 6 bis 8 enthalten wird.
Für das obige Beispiel, wenn m = 3 und die Vorlage B = 001, dann die Prüfung läuft wie folgt:

Bitpositionen
Block 1 Block 2
Bits W1 Bits W2
1-3 101 0 111 0
2-4 010 0 110 0
3-5 100 0 100 0
4-6 001 (hit) Increment auf 1 001 (hit) Increment auf 1

5-7 nicht untersucht nicht untersucht
6-8 nicht untersucht nicht untersucht
09.07 001 Increment zu 2 011 1
80-10 010 (hit) 2 110 1

So W1 = 2, und W2 = 1 ist.

(3)
Unter der Annahme, des Zufalls, berechnen den theoretischen Mittelwert μ und Varianz s2:
μ = (M-m +1) / 2m
2-15

= (
.

Analyse

Für das Beispiel in diesem Abschnitt μ = (10-3 +1) / 23 = 1, und
(
.

(4) Compute.
obs (
$
Für das Beispiel in diesem Abschnitt

".
(5) Berechnen Sie P-Wert = igamc $ $%
&
, 2N2. Beachten Sie, dass mehrere P-Werte berechnet wird, dh,
ein P-Wert wird für jede Vorlage berechnet werden. Für m = 9, bis zu 148 P-Werte können
berechnet, für m = 10, bis zu 284 P-Werten berechnet werden.

Für das Beispiel in diesem Abschnitt, P-Wert = igamc

$%
&
0,344154.

2.7.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.7.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 5 des Abschnitts 2.7.4 erhält man =

0,01 (p-Wert = 0,344154), ist die Schlussfolgerung, dass
die Reihenfolge ist zufällig.

Wenn der P-Wert ist sehr klein (<0,01), dann ist die Folge hat unregelmäßige Auftreten der möglichen Vorlage
Mustern.

2.7.7 Input Size Empfehlung
Der Test-Code wurde geschrieben, um Vorlagen für m = 2, 3, ..., 10 vorzusehen. Es wird empfohlen, m = 9 oder
m = 10 angegeben werden, um aussagekräftige Ergebnisse zu erhalten. Obwohl N = 8 in der Test-Code angegeben wurde, die
Code kann auf andere Größen verändert werden. Allerdings sollte N so gewählt, dass N = werden
100 bis sichergestellt werden, dass die
P-Werte sind gültig. Der Test-Code wurde geschrieben, um eine Sequenzlänge von n = 106 (über eine Eingabe übernehmen
Aufruf-Parameter) und M = 131072 (hart codiert). Bei anderen Werten als diese gewünscht sind, sicher sein, dass M>

0,01 • n und N =. N / M..
2.7.8 beispiel
Für eine Vorlage B = 000000001, deren Größe m = 9:
(Eingang) e
= 220 Bits, die durch die G-SHA-1 Generator1 produziert
(Eingang) n = 220, B = 000000001
(Verarbeitung) μ = 255,984375 und s2 = 247,499999

1 in Federal Information Processing Standard (FIPS) 186-2 definiert.

Analyse

(Verarbeitung) W1 = 259; W2 = 229; W3 = 271; W4 = 245; W5 = 272; W6 = 262; W7 = 259; und W8 = 246
(Verarbeitung) .2 (obs) = 5,999377
(Output) P-Wert = 0,647302
(Schluss) Da die P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,8 Overlapping Template Matching-Test
2.8.1 Test-Purpose
Der Schwerpunkt der Overlapping Template Matching-Test ist die Anzahl der Vorkommen des vorgegebenen Ziels
Saiten. Beide dieser Prüfung und den Non-überlappenden Template Matching Test der

Abschnitt 2.7 verwenden Sie ein m-Bit-
Fenster, um für eine bestimmte m-Bit-Muster zu suchen. Wie bei dem Test in Abschnitt 2.7, wenn das Muster nicht gefunden wird,
das Fenster gleitet eine Bitposition. Der Unterschied zwischen diesem Test und der Test in Abschnitt 2.7 ist, dass
wenn das Muster gefunden wird, gleitet das Fenster nur ein Bit vor der Wiederaufnahme der Suche.

2.8.2 Function Call
OverlappingTemplateMatching (m, n)
m Die Länge in Bits der Vorlage - in diesem Fall die Länge des Laufes von Einsen.
n Die Länge des Bit-String.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:
e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als

globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.
B Die m-Bit-Vorlage abgestimmt werden.
K Die Zahl der Freiheitsgrade. K hat bei 5 in der Test-Code behoben.
M Die Länge in Bits eines Teilstrings von e
getestet werden. M hat bis 1032 wurden in den Test-Code festgelegt.
N Die Anzahl der unabhängigen Blöcken von n. N hat bis 968 wurden in den Test-Code festgelegt.

2.8.3 Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachtete Anzahl der Vorlage "hits" entspricht der erwarteten
Anzahl der Vorlage "Hits" (unter der Annahme des Zufalls).

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.8.4 Test Beschreibung
(1) Partition der Folge in N unabhängige Blöcke der Länge M.

Analyse

Zum Beispiel, wenn e
= 10111011110010110100011100101110111110000101101001, dann n = 50.
Wenn K = 2, M = 10 und N = 5, dann die fünf Blöcke sind 1011101111, 0010110100, 0111001011,
1011111000, 0101101001 und.

(2)
Berechnen Sie die Anzahl der Vorkommen von B in jedem der N Blöcke. Die Suche nach Übereinstimmungen
Erlöse durch die Schaffung eines m-Bit-Fenster auf die Reihenfolge, vergleicht die Bits in diesem Fenster
gegen B und Erhöhen eines Zählers, wenn es eine Übereinstimmung gibt. Das Fenster gleitet über ein Bit nach
jede Untersuchung, zB wenn m = 4 und das erste Fenster enthält Bits 42 bis 45, im nächsten Fenster
besteht aus Bits 43 bis 46. Notieren Sie die Anzahl der Vorkommen von B in jedem Block durch Inkrementieren
eine Reihe vi (i = 0, ... 5), so dass v0 erhöht, wenn es keine Vorkommen von B in A ist substring, v1 ist für ein Vorkommen von B erhöht, ... und v5 ist für 5 oder mehr erhöht Vorkommen von B.
Für das obige Beispiel, wenn m = 2 und B = 11, dann die Prüfung des ersten Blocks (1011101111)
läuft wie folgt ab:

Bitpositionen Bits Anzahl der Vorkommen von B = 11
1-2 10 0
2-3 01 0
03-04 November (hit) Increment auf 1
November 04-05 (hit) Increment bis 2
5-6 10 2
6-7 01 2
07-08 November (hit) Increment bis 3
NOVEMBER 08 bis 09 (Treffer) Increment bis 4
November von 09 bis 10 (Treffer) Increment bis 5

So, nach Block 1 gibt es fünf Vorkommen von 11 ist v5 erhöht, und v0 = 0, v1 = 0, v2 = 0,
v3 = 0, v4 = 0 und v5 = 1 ist.

In gleicher Weise werden die Blöcke 2-5 untersucht. In Block 2, gibt es 2 Vorkommen von 11; v2 ist
erhöht. In Block 3 gibt es 3 Vorkommen von 11; v3 wird erhöht. In Block 4 gibt es 4 Vorkommen von 11; v4 wird erhöht. In Block 5, gibt es ein Vorkommen von 11; v1 wird erhöht.

Daher v0 = 0, v1 = 1, v2 = 1, v3 = 1, v4 = 1, v5 = 1 nach allen Blöcken untersucht worden sind.

(3)
Compute-Werte für.
und.
das wird benutzt, um die theoretischen Wahrscheinlichkeiten pi zu berechnen entsprechend den Klassen von v0:
.

= (M-m +1) / 2m.
=. / 2.

Für das Beispiel in diesem Abschnitt.
= (10-2 +1) / 22 = 2,25, und.
=. / 2 = 1,125.

(4)
Berechnen
$
, Wo p0 = 0,364091, p1 = 0,185659, p2 = 0,139381, p3
= 0.100571, p4 = 0,070432 und p5 = 0,139865 wie in Abschnitt 3.8 angegeben.
Für das Beispiel in diesem Abschnitt wurden die Werte von pi neu berechnet, da das Beispiel nicht passt
die genannten Anforderungen in Abschnitt 2.8.7. Das Beispiel ist nur zur Illustration gedacht. Die Werte

Analyse

von pi sind: p0 = 0,324652, p1 = 0,182617, p2 = 0,142670, p3 = 0,106645, p4 = 0,077147 und p5 =

0,166269. "(
! = 3,167729.
(5) Berechnen Sie P-Wert = igamc
$ $%
&
, 252.
Für das Beispiel in diesem Abschnitt, P-Wert = igamc $%
&
0,274932.
2.8.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.8.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 4 des Abschnitts 2.8.4 erhält man =
0,01 (p-Wert = 0,274932), ist die Schlussfolgerung, dass
die Reihenfolge ist zufällig.

Beachten Sie, dass für die 2-Bit-Vorlage (B = 11), wenn die gesamte Sequenz zu viele 2-Bit läuft von Einsen, dann: 1)

0,5 wäre zu groß, 2) haben die Teststatistik zu groß wäre, 3) die P-Wert gewesen wäre klein (<0,01) und 4) eine Schlussfolgerung einer Nicht-Zufälligkeit geführt hätte.

2.8.7 Input Size Empfehlung
Die Werte von K, M und N wurden so gewählt, dass jede Sequenz zu prüfenden von mindestens besteht
von 106 Bit (also, n =
106). Verschiedene Werte von m gewählt werden kann, aber für den Augenblick, empfiehlt NIST
m = 9 oder m = 10. Wenn andere Werte gewünscht werden, wählen Sie bitte diese Werte wie folgt:

•
n =
MN.
•
N sollte so gewählt werden, so dass N • (min pi)> 5.
•
.
= (M-m +1) / 2m ~
2
•
m sollte, so dass m gewählt werden ~
log2 M
•
Wählen Sie K, so dass K ~
2 .. Beachten Sie, dass der PI-Werte müssten für Werte neu berechnet werden
K andere als 5.
2.8.8 beispiel
(Eingang) e
= Das binäre Ausbau von E bis zu 1.000.000 Bits

(Eingang) n = 1000000, B = 111111111

(Verarbeitung) .0 = 329; .1 = 164; .2 = 150; .3 = 111; .4 = 78; und .5 = 136

(Verarbeitung) .2 (obs) = 8,965859

(Output) P-Wert = 0,110434

2-19

Analyse

(Schluss) Da die P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,9 Maurers "Universal Statistical" Test

2.9.1 Testzwecken

Der Fokus dieses Tests ist die Anzahl der Bits zwischen übereinstimmenden Muster (eine Maßnahme, die verwandt ist
Länge eines komprimierten Sequenz). Der Zweck des Tests ist zu erkennen, ob die Sequenz kann
deutlich, ohne Verlust von Informationen komprimiert. Eine deutlich kompressiblen Sequenz ist
als nicht-zufällig.

2.9.2 Function Call

Universal (L, Q, n), wo

L
Die Länge der einzelnen Blöcke. Hinweis: die Verwendung von L als die Blockgröße ist nicht konsistent mit der
Blockgröße Notation (M) für die anderen Tests verwendet. Allerdings ist die Verwendung von L als die Blockgröße
war in der ursprünglichen Quelle der Maurer-Test angegeben.

Q
Die Anzahl der Blöcke in der Initialisierungssequenz.

n
Die Länge des Bit-String.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.9.3 Teststatistik und Reference Vertrieb

fn:
Die Summe der Abstände zwischen log2 passenden L-Bit-Vorlagen, dh die Summe der Anzahl der
Ziffern in den Abstand zwischen L-Bit-Vorlagen.

Die Referenz-Verteilung für die Teststatistik ist die Halb-Normalverteilung (eine einseitige Variante des
Normalverteilung), wie es auch der Fall für die Frequenz-Test in Abschnitt 2.1.

2.9.4

Test Beschreibung
(1)
Die n-Bit-Sequenz (e) wird in zwei Segmente unterteilt: eine Initialisierung Segment, bestehend aus Q
L-Bit nicht überlappende Blöcke, und ein Test-Segment, bestehend aus
KL-Bit-nicht-überlappenden Blöcken.
Bits übrigen am Ende der Sequenz, die nicht zur einer kompletten L-Bit-Block werden verworfen.
Die ersten Q-Blöcke werden verwendet, um den Test zu initialisieren. Die verbleibenden K-Blöcke sind die Test-Blöcke (K =
. N / L.
-Q).
2-20

Analyse

Zum Beispiel, wenn e
= 01011010011101010111, dann n = 20. Wenn L = 2 und Q = 4, dann K =. N / L.
Q
= .20 / 2.
-4 = 6. Die Initialisierung Segment 01011010; der Test-Segment ist 011101010111.
Die L-Bit-Blöcke sind in der folgenden Tabelle dargestellt:

Block Type Inhalt
1 01
2 Initialisierung 01
3 Segment 10
4 10
5 01
6 Test Segment 11
7 01
8 01
9 01
10 11

(2)
Mit der Initialisierung Segment ist eine Tabelle für jede mögliche L-Bit-Wert (dh die L-Bit erstellt
Wert wird als Index in der Tabelle verwendet wird). Die Satznummer des letzten Auftretens der einzelnen L-Bit-
Block ist in der Tabelle vermerkt (dh für i von 1 bis Q, Tj = i, wobei j der Dezimaldarstellung
den Inhalt des i-ten L-Bit-Block).
Für das Beispiel in diesem Abschnitt wird die folgende Tabelle erstellt mit den 4-Initialisierung blockiert.

Mögliche L-Bit-Wert
00
(Gespeichert in T0)
01
(Gespeichert in T1)
10
(Gespeichert in T2)
11
(Gespeichert in T3)
Initialization 0 2 4 0

(3)
Untersuchen Sie jede der K-Blöcke in den Test-Segment und die Anzahl der Blöcke, da die
letzte Vorkommen des gleichen L-Bit-Block (dh i - Tj). Ersetzen Sie den Wert in der Tabelle mit den
Standort des aktuellen Blocks (also, Tj = i). Fügen Sie die berechnete Abstand zwischen re-Vorkommen
die gleiche L-Bit-Block zu einer Anhäufung log2 Summe aller Unterschiede in den K-Sohlen erkannt
(D. h. sum = sum + log2 (i - Tj)).
Für das Beispiel in diesem Abschnitt werden die Tabelle und die kumulative Summe wie folgt entwickelt:

Für Block 5 (1. Test-Block): 5 in der "01" Zeile der Tabelle (dh T1) gelegt, und
sum = log2 (5-2) = 1,584962501.
Für Block 6: 6 in der "11" Zeile der Tabelle (dh, T3), und die Summe platziert = 1,584962501 +
log2 (6-0) = 1,584962501 + 2,584962501 = 4,169925002.
Für Block 7: 7 in der "01" Zeile der Tabelle (dh T1), und die Summe platziert = 4,169925002 +
log2 (7-5) = 4,169925002 + 1 = 5,169925002.
Für Block 8: 8 ist in der "01" Zeile der Tabelle (dh T1) gelegt, und sum = 5,169925002 +
log2 (8-7) = 5,169925002 + 0 = 5,169925002.
Für Block 9: 9 in der "01" Zeile der Tabelle (dh T1), und die Summe platziert = 5,169925002 +
log2 (9-8) = 5,169925002 + 0 = 5,169925002.

Analyse

Für Block 10: 10 in der "11" Zeile der Tabelle (dh, T3), und die Summe platziert = 5,169925002

+ Log2 (10-6) = 5,169925002 + 2 = 7,169925002.

Die Zustände der Tabelle sind:

Iteration Mögliche L-Bit-Wert
Block 00 01 10 11
4 0 2 4 0
5 0 5 4 0
6 0 5 4 6
7 0 7 4 6
8 0 8 4 6
9 0 9 4 6
10 0 9 4 10

(4) Berechnen Sie die Teststatistik:, wo =
wird die Tabelle entsprechende Eintrag
die dezimale Darstellung der Inhalte der i-ten L-Bit-Block.
Für das Beispiel in diesem Abschnitt

= 1,1949875.

(5) Berechnen Sie P-Wert = erfc $ $%
& LueexpectedVafn, wo erfc in Abschnitt 5.5.3 definiert ist, und
expectedValue (L) und s
sind aus einer Tabelle mit vorberechneten Werte2 (siehe Tabelle unten) berücksichtigt.
Unter der Annahme, des Zufalls, bedeutet der Probe wird expectedValue (L), die
theoretische

erwarteten Wert der berechneten Statistik für den gegebenen L-Bit Länge. Die
theoretische Norm
Abweichung ist gegeben durch!, wo =
.
L expectedValue Varianz
6 5.2177052 2,954
7 6.1962507 3,125
8 7.1836656 3,238
9 8.1764248 3,311
10 9.1723243 3,356
11 10.170032 3,384

L expectedValue Varianz
12 11.168765 3,401
13 12.168070 3,410
14 13.167693 3,416
15 14.167488 3,419
16 15.167379 3,421

Für das Beispiel in diesem Abschnitt, P-Wert = erfc

$ $%

& Hinweis
dass der Erwartungswert und Varianz für L = 2 sind nicht in der obigen Tabelle angegeben, da ein Block von
Länge zwei ist nicht zur Prüfung empfohlen. Allerdings ist dieser Wert für L leicht in einer Anwendung

2 Aus dem "Handbook of Applied Cryptography".

Analyse

Beispiel. Der Wert für den Erwartungswert und Varianz für den Fall L = 2, wenn auch nicht
gezeigt, in der obigen Tabelle wurden aus den angegebenen Reference3 genommen.

2.9.5 Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.9.6 Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 5 des Abschnitts 2.9.4 erhält man =
0,01 (p-Wert = 0,767189), ist die Schlussfolgerung, dass
die Reihenfolge ist zufällig.

Theoretische Erwartungswerte für.
wurden berechnet wie in der Tabelle in Schritt (5) von Abschnitt 2.9.4 gezeigt.
Wenn fn unterscheidet sich deutlich von expectedValue (L), so wird der Ablauf wesentlich komprimierbar.

2.9.7 Input Size Empfehlung
Dieser Test erfordert eine lange Folge von Bits (n =
(Q + K) L), die in zwei Segmenten der L-Bit aufgeteilt werden
Blöcke, in denen L gewählt werden sollte, so dass 6 =
L =
16. Das erste Segment besteht aus Q-Initialisierung blockiert,
wo Q gewählt werden sollte, so dass Q = 10 • 2L. Das zweite Segment besteht aus K-Test Blöcke, wobei K =
. N / L.
-Q ~
1000 • 2L. Die Werte von L, Q und n wie folgt gewählt werden:

n L Q = 10 • 2L
=
387,840 6 640

=
904,960 7 1280
=
2,068,480 8 2560
=
4,654,080 9 5120
=
1,342,400 10 10240
=
22,753,280 11 20480
=
49,643,520 12 40960
=
107,560,960 13 81920
=
231,669,760 14 163840
=
496,435,200 15 327680
=
1,059,061,760 16 655360

2.9.8 beispiel
(Eingang) e
= A Binärstring unter Verwendung G-SHA-14
(Eingang) n = 1048576, L = 7, Q = 1280
(Hinweis) Hinweis: 4 Bits werden verworfen.
(Verarbeitung) c = 0,591311, s
= 0.002703, K = 148516, sum = 919924,038020
(Verarbeitung) fn = 6,194107, expectedValue = 6,196251, s
= 3,125
(Output) P-Wert = 0,427733

3 Aus dem "Handbook of Applied Cryptography".
4 in FIPS 186-2 definiert.

Analyse

(Schluss)
Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,10 lineare Komplexität Testen
2.10.1
Testzwecken

Der Fokus dieses Tests ist die Länge eines Linear Feedback Shift Register (LFSR). Der Zweck dieses Tests ist es,
festzustellen, ob die Sequenz komplex genug, um als zufällig ist. Zufällige Folgen werden von mehr LFSRs aus. Ein LFSR zu kurz ist, bedeutet nicht Beliebigkeit.

2.10.2
Function Call
LinearComplexity (M, n), wobei:

M
Die Länge in Bits eines Blocks.

n Die Länge des Bit-String.
Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.
K Die Zahl der Freiheitsgrade; K = 6 wurde hart in die Test-codiert.

2.10.3
Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachteten Anzahl von Vorkommen fester Länge LFSRs entspricht der erwarteten Anzahl von Vorkommen unter einer Annahme des Zufalls.

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.10.4
Test Beschreibung
(1)
Partition der n-Bit-Sequenz in N unabhängigen Blöcken von M Bits, wobei n = MN.
(2)
Mit der Berlekamp-Massey algorithm5, die lineare Komplexität Li jedes der N Blöcke (i = 1, ..., N). Li ist die Länge des kürzesten Linear Feedback Shift Register Sequenz,
erzeugt alle Bits im Block i. Innerhalb einer Li-Bit-Sequenz, eine Kombination von Bits, wenn
addiert modulo 2, produziert das nächste Bit in der Folge (bit Li + 1).
Zum Beispiel, wenn M = 13 und der Block der getestet werden soll 1101011110001, dann Li = 4, und die
Sequenz wird durch Addition der 1. und 2. Bits innerhalb eines 4-Bit-Teilfolge zum nächsten herstellen können
bit (5. bit). Die Untersuchung verlief wie folgt:

A. Menezes, P. Van Oorschot und S. Vanstone;; CRC Press, 1997 5 in The Handbook of Applied Cryptography definiert.

Analyse

Bit 1 Bit 2 Bit 3 Bit 4 Bit 5
Die ersten 4 Bit und die daraus resultierenden 5. Bit 1 1 0 1 0
Bits 2-5 und die daraus resultierenden 6. Bit 1 0 1 0 1
Bits 3-6 und die daraus resultierende 7. Bit 0 1 0 1 1
Bits 4-7 und die daraus resultierenden 8. Bit 1 0 1 1 1
Bits 5-8 und die daraus resultierende 9. Bit 0 1 1 1 1
Bits 6-9 und dem daraus resultierenden 10. Bit 1 1 1 1 0
Bits 7-10 und die daraus resultierenden 11. Bit 1 1 1 0 0
Bits 8-11 und die daraus resultierenden 12. Bit 1 1 0 0 0
Bits 9-12 und die daraus resultierenden 13. Bit 1 0 0 0 1

Aus diesem Block arbeitet die Studie Feedback-Algorithmus. Wenn dies nicht der Fall wäre, andere Rückmeldungen
Algorithmen würden für den Block (z. B. versucht werden, indem die Bits 1 und 3 bis Bit 5 zu produzieren, oder das Hinzufügen von
Bits 1, 2 und 3 bis Bit 6, etc. zu erzeugen).

(1)
Unter der Annahme, des Zufalls, die Berechnung der theoretischen Mittelwert μ:
+ =
μ.
Für das Beispiel in diesem Abschnitt
+ =
μ = 6,777222.
Für jeden Teilstring, berechnen einen Wert von Ti, wo = μ.
(4)
Für das Beispiel = (5)
Notieren Sie sich die Ti-Werte in v0, ..., v6 wie folgt:
Wenn:
Ti =
-2,5 Increment vo durch eine
-2,5 <Ti =
-1,5 Increment v1 von einem
-1,5 <Ti =
-0,5 Increment v2 von einem
-0,5 <Ti =
0,5 Increment v3 von einem

0,5 <Ti =
1,5 Increment v4 von einer

1,5 <Ti =
2,5 Increment v5 durch eine
Ti> 2,5 Increment v6 von einem
(6) Compute
obs (
$
, Wo p0 = 0,01047, p1 = 0,03125, p2 = 0,125, p3 = 0,5, p4 =
0,25, p5 = 0,0625, p6 = 0,02078 werden die Wahrscheinlichkeiten durch die Gleichungen in Abschnitt 3.10 berechnet.

(7)
Berechnen Sie P-Wert = igamc
$ $%
&
, 2K2.
2.10.5
Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

Analyse

2.10.6 Zusammenfassung und Interpretation der Ergebnisse
Wenn der P-Wert wurden <0,01, würde dies darauf hin, dass die beobachteten Häufigkeiten der Ti in gespeichert haben
. die ich bins von den erwarteten Werten variiert, es wird erwartet, dass die Verteilung der Häufigkeit der Ti
(In der. Ich bins) sollte proportional zu den berechneten pi wie in Schritt (6) von Abschnitt 2.10.5 gezeigt.

2.10.7 Input Size Empfehlung
Wählen Sie n =
106. Der Wert der M muss im Bereich von 500 werden =
M =
5000 und N =
200 für den 0,2 Ergebnis zu
gültig ist (siehe Abschnitt 3.10 für eine Diskussion).

2.10.8 beispiel
(Eingang) e
= "Die ersten 1.000.000 binären Ziffern in den Ausbau des e"
(Eingang) n = 1000000 = 106, M = 1000
(Verarbeitung) v0 = 11; v1 = 31; v2 = 116; v3 = 501; v4 = 258; v5 = 57; v6 = 26

(Verarbeitung) .2 (obs) = 2,700348
(Output) P-Wert = 0,845406
(Schluss) Da die P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,11 Seriell-Test
2.11.1 Test-Purpose
Der Fokus dieses Tests ist die Häufigkeit aller möglichen Überschneidungen m-Bit-Muster über die gesamte
Sequenz. Der Zweck dieses Tests ist es festzustellen, ob die Anzahl der Vorkommen des 2mm-Bit-
überlappende Muster ist ungefähr die gleiche, wie für eine zufällige Sequenz zu erwarten. Zufällig
Sequenzen Einheitlichkeit, das heißt, hat jeder m-Bit-Muster die gleiche Chance zu erscheinen, wie jedes andere
m-Bit-Muster. Beachten Sie, dass für m = 1, die Serial-Test entspricht der Frequenz-Test von Abschnitt 2.1 ist.

2.11.2 Function Call
Serial (m, n), wobei:

m Die Länge in Bits pro Block.

n Die Länge in Bits des Bit-String.

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:
e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.11.3 Teststatistik und Reference Vertrieb
.. 2m (obs) und.
2,2 m (obs):
Ein Maß dafür, wie gut die beobachteten Häufigkeiten von m-Bit-Muster entsprechen die erwarteten Frequenzen der m-Bit-Mustern.

Analyse

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.11.4 Test Beschreibung
(1)
Form eines Augmented-Sequenz e ': Erweitern Sie die Sequenz, die durch das

Anhängen der ersten m-1-Bits bis zum Ende
der Sequenz für verschiedene Werte von n.
Zum Beispiel, n = 10 und e gegeben
= 0011011101. Wenn m = 3, dann e '= 001101110100. Wenn m = 2,
dann e '= 00110111010. Wenn m = 1, dann e '= die ursprüngliche Reihenfolge
0011011101.

(2)
Bestimmen Sie die Frequenz aller möglichen überlappenden
1m1i ... iv
m-Bit-Blöcken, die alle möglichen Überlappung (
m1i ... ivm-1)-Bit-
Blöcke und alle möglichen Überschneidungen (m-2)-Bit-Blöcke. Lassen Sie bezeichnen die Frequenz der m-
Bitmuster i1 ... im, lassen Sie bezeichnen die Frequenz der (m-1)-Bit-Muster i1 ... im-1, und lassen

2m1i ... iv
bezeichnen die Frequenz der (m-2)-Bit-Muster i1 ... im-2.
Für das Beispiel in diesem Abschnitt, wenn m = 3, dann ist (m-1) = 2, und (m-2) = 1 ist. Die Häufigkeit aller
3-Bit-Blöcke: V000 = 0, v001 = 1, V010 = 1, V011 = 2, v100 = 1, v101 = 2, v110 = 2, V111 = 0 ist. Sterben
Häufigkeit aller möglichen (m-1)-Bit-Blöcke ist: v00 = 1, v01 = 3, v10 = 3, v11 = 3. Die Häufigkeit aller
(M-2)-Bit-Blöcke ist: v0 = 4, v1 = 6.

(3) Berechnen Sie: =
) "
) "
)
Für das Beispiel in diesem Abschnitt

Für das Beispiel in diesem Abschnitt

0,2
3 =
1023 =. 2
2 =
1022 =. 2
1 =
102 = (4) Berechnen Sie:
# Und!
#.
#$#$(5) Berechnen Sie: P-Wert1 = igamc
$% &
(2und
2-27

Analyse

P-Wert2 = igamc

$% &
(2.
Für das Beispiel in diesem Abschnitt
P-Wert1 = igamc $%
&
$%
&
0,9057

P-Wert2 = igamc

0,8805.

2.11.5 Entscheidung Rule (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.11.6 Zusammenfassung und Interpretation der Ergebnisse
Da die P-Wert in Schritt 5 des Abschnitts 2.11.4 gewonnen wird =
0,01 (P-Wert1 = 0,808792 und P-Wert2 =
0.670320), ist die Schlussfolgerung, dass die Reihenfolge zufällig ist.

Beachten Sie, dass, wenn .2.2 m oder .. 2m hatte großes dann Uneinheitlichkeit der m-Bit-Blöcke impliziert ist.

2.11.7 Input Size Empfehlung
Wählen m und n, so dass m <. Log2 n.
-2.

2.11.8 beispiel
(Eingang) e
= 1.000.000 Bits aus dem binären Ausbau der e
(Input) m = 2, n = 1000000 = 106
(Verarbeitung) # 0s = 499971; # 1s = 500029

00s = 250116; # 01s = # 10s = 249855; # 11s = 250174
(Verarbeitung) .22 = 0,343128; 0,21 = 0,003364; 0,20 = 0,000000
(Verarbeitung) .. 22 = 0,339764; .2.22 = 0,336400
(Output) P-Wert1 = 0,843764; P-Wert2 = 0,561915

(Schluss)
Da beide P-Wert1 und P-value2 wurden =
0,01, akzeptieren die Sequenzen als zufällig für beide
Tests.

2,12 Ungefähre Entropy-Test
2.12.1 Test-Purpose
Wie bei den Serial-Test von Abschnitt 2.11, liegt der Fokus dieses Tests die Häufigkeit aller möglichen überlappenden
m-Bit-Muster über die gesamte Sequenz. Der Zweck des Tests ist es, die Häufigkeit der zu vergleichen
überlappende Blöcke von zwei aufeinander folgenden / angrenzende Längen (m und m +1) gegen das erwartete Ergebnis für eine
zufälliger Reihenfolge.

Analyse

2.12.2
Function Call
ApproximateEntropy (m, n), wobei:
m Die Länge der einzelnen Blöcke - in diesem Fall der erste Block Länge in den Test eingesetzt. m +1 ist die
zweiten Block Länge verwendet.
n Die Länge der gesamten Bitfolge.
Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:
e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.12.3
Teststatistik und Reference Vertrieb
.2 (Obs):
Ein Maß dafür, wie gut die beobachteten Wert von ApEn (m) (siehe Schritt 6 im Abschnitt 2.12.4)
entspricht dem erwarteten Wert.

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.12.4
Test Beschreibung
(1)
Erweitern Sie das n-Bit-Sequenz zu erstellen n überlappenden m-Bit-Sequenzen durch Anhängen von m-1 Bits aus

Beginn der Sequenz bis zum Ende der Sequenz.
Zum Beispiel, wenn e
= 0100110101 und m = 3, dann n = 10. Fügen Sie den 0 und 1 zu Beginn des
Die Sequenz für das Ende der Sequenz. Die Sequenz, die getestet werden wird
010011010101.
(Hinweis: Dies ist für jeden Wert von m. getan)

(2)
Eine Frequenz count ist der n überlappende Blöcke (zB aus, wenn ein Block mit ej zu ej + m-1 ist
untersucht zur Zeit j, dann wird der Block mit ej +1 bis ej + m ist an der Zeit Cmij 1 untersucht). Lassen Sie die Anzahl
der möglichen m-bit ((m +1)-Bit)-Werte als, wobei i die m-Bit-Wert dargestellt werden.

Für das Beispiel in diesem Abschnitt werden die überlappenden m-Bit-Blöcke (mit m = 3) 010, 100,
001, 011, 110, 101, 010, 101, 010 und 101. Die berechneten zählt für die 2m = 23 = 8 möglichen m-
Bit-Strings sind:

000 = 0, # 001 = 1, # 010 = 3, # 011 = 1, # 100 = 1, # 101 = 3, # 110 = 1, Nr. 111 = 0

(3) für jeden Wert von i. Compute

Zum Beispiel in diesem Abschnitt, CMI =
C3000 = 0, C3001 = 0,1, C3010 = 0,3, C3011 = 0,1, C3100 = 0,1, C3101 = 0,3, C3110 = 0,1, C3111 = 0 ist.

(4) Berechnen Sie
m (
$
, Wo pi = C3j und j = log2 i.
Für das Beispiel in diesem Abschnitt. (3) = 0 (log 0) + 0,1 (log 0,1) + 0,3 (log 0,3) + 0,1 (log 0,1) +
0,1 (log 0,1) + 0,3 (log 0,3) + 0,1 (log 0,1) +0 (log 0) = -1,64341772.

Analyse

.
(5) Wiederholen Sie die Schritte 1-4, ersetzen m durch m +1.
Schritt 1: Für das Beispiel in diesem Abschnitt ist jetzt 4 m, wobei die Reihenfolge getestet werden wird
0100110101010.

Schritt 2: Die überlappenden Blöcke werden 0100, 1001, 0011, 0110, 1101, 1010, 0101, 1010, 0101,
1010. Die berechneten Werte sind: # 0011 = 1, # 0100 = 1, # 0101 = 2, # 0110 = 1, # 1001 = 1, # 1010
= 3, # 1101 = 1, und alle anderen Muster sind gleich Null.

Schritt 3: C40011 C40100 = = = C40110 C41001 C41101 = = 0,1, C40101 = 0,2, C41010 = 0,3, und alle anderen
Werte sind gleich Null.

Schritt 4:. (4) = 0 + 0 + 0 + 0,1 (log 0,01) + 0,1 (log 0,01) + 0,2 (log 0,02) + 0,1 (log 0,01) + 0 + 0 +
0,1 (log 0,01) + 0,3 (log 0,03) +0 +0 +0,1 (log 0,01) + 0 + 0) = -1,83437197.

(6) Berechnen Sie die Teststatistik: .2 = 2n [log 2 - Apen (m)], wobei
Für das Beispiel in diesem Abschnitt

ApEn (3) = -1,643418 - (-1,834372) = 0,190954
2 = 2 • 10 (0.693147-0.190954) = 0,502193

(7) Berechnen Sie P-Wert = igamc (2m-1,
22!
).
Für das Beispiel in diesem Abschnitt, P-Wert = igamc

$%
&
= 0,261961.

2.12.5 Entscheidung Rule (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.12.6 Zusammenfassung und Interpretation der Ergebnisse
Da die P-Wert in Schritt 7 des Abschnitts 2.12.4 gewonnen wird =
0,01 (p-Wert = 0,261961), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass kleine Werte von ApEn (m) würde starke Regelmäßigkeit impliziert (siehe Schritt 6 des Abschnitts 2.12.4). Brutto
Werte bedeuten würde erhebliche Fluktuation oder Unregelmäßigkeiten.

2.12.7 Input Size Empfehlung
Wählen m und n, so dass m <. Log2 n.
-2.

2.12.8 beispiel
(Eingang) e
= 1100100100001111110110101010001000100001011010001100001000110100110001001100011001100010100001011100 0
(Input) m = 2, n = 100
(Verarbeitung) ApEn (m) = 0,665393
(Verarbeitung) .2 (obs) = 5,550792

Analyse

(Output) P-Wert = 0,235301
(Schluss) Da P-Wert =
0,01, akzeptieren die Sequenz als zufällig.

2,13 Summenwerte (Cusum) Test
2.13.1
Testzwecken
Der Fokus dieses Tests ist die maximale Auslenkung (von Null) der Irrfahrt durch die kumulative definiert
Summe der berichtigten (-1, +1) Stellen in der Sequenz. Der Zweck der Prüfung soll festgestellt werden, ob die
kumulative Summe der Teilsequenzen auftreten in den getesteten Sequenz ist zu groß oder zu klein im Verhältnis
um das erwartete Verhalten, dass eine kumulative Summe für zufälligen Sequenzen. Dieses kumulative Summe kann
gilt als einer Irrfahrt. Für eine zufällige Folge sollte die Exkursionen der Irrfahrt in der Nähe sein
Null. Für bestimmte Arten von nicht-zufälligen Sequenzen, werden die Exkursionen dieser Irrfahrt von Null werden
groß ist.

2.13.2
Function Call
CumulativeSums (Modus, n), wobei:
n Die Länge des Bit-String.
Zusätzlicher Eingang für die Funktion, sondern versorgt die Testing-Code:
e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.
Modus A-Schalter für die Anwendung der Test entweder nach vorne durch die Input-Sequenz (mode = 0) oder
rückwärts durch die Sequenz (mode = 1).

2.13.3
Teststatistik und Reference Vertrieb
z:
Der größte Ausflug von der Herkunft der kumulativen Summen in die entsprechenden (-1, +1)
Sequenz.
Die Referenz-Verteilung für die Teststatistik ist die Normalverteilung.

2.13.4
Test Beschreibung
(1)
Form einer normalisierten Folge: Die Nullen und Einsen der Eingabesequenz (e) sind umgesetzt
Werte Xi von -1 und +1 mit Xi = 2ei - 1.
Zum Beispiel, wenn e
= 1011010111, dann X = 1, (-1), 1, 1, (-1), 1, (-1), 1, 1, 1.

(2)
Compute Partialsummen Si sukzessive größer Teilfolgen jeweils beginnend mit X1 (wenn mode = 0)
oder Xn (wenn mode = 1).

Analyse

Mode = 0 (vorwärts) Mode = 1 (rückwärts)
S1 = X1
S2 = X1 + X2
S3 = X1 + X2 + X3
.
.
Sk = X1 + X2 + X3 + ... + Xk
.
.
Sn = X1 + X2 + X3 + ... + Xk + ... + Xn
S1 = Xn
S2 = Xn + Xn-1
S3 = Xn + Xn-1 + Xn-2
.
.
Sk = Xn + Xn-1 + Xn-2 + ... + Xn-k +1
.
.
Sn = Xn + Xn-1 + Xn-2 + ... + Xk-1 + ... + X1

Das heißt, Sk = Sk-1 + Xk für Modus 0 und Sk = Sk-1 + Xn-k +1 für mode 1.

Für das Beispiel in diesem Abschnitt, wenn mode = 0 und X = 1, (-1), 1, 1, (-1), 1, (-1), 1, 1, 1, dann gilt:

S1 = 1
S2 = 1 + (-1) = 0
S3 = 1 + (-1) + 1 = 1
S4 = 1 + (-1) + 1 + 1 = 2
S5 = 1 + (-1) + 1 + 1 + (-1) = 1
S6 = 1 + (-1) + 1 + 1 + (-1) + 1 = 2
S7 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) = 1
S8 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 = 2
S9 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + 1 = 3
S10 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + 1 + 1 = 4

(3)
Berechnen Sie die Prüfgröße z = max1 = k = n | Sk |, wo max1 = k = n | Sk | ist die größte der absoluten Werte der
die Teilsummen Sk.
Für das Beispiel in diesem Abschnitt ist der größte Wert von Sk 4, so z = 4.

(4) Berechnen Sie P-Wert =
(
!
% &
"
(
=
wo F
ist die Standardnormalverteilung Verteilungsfunktion im Sinne
Abschnitt 5.5.3.

Für das Beispiel in diesem Abschnitt, P-Wert = 0,4116588.

2.13.5
Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist.
Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.13.6 Zusammenfassung und Interpretation der Ergebnisse
Da die P-Wert in Schritt 4 des Abschnitts 2.13.4 gewonnen wird =
0,01 (p-Wert = 0,411658), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass, wenn mode = 0, große Werte von dieser Statistik zeigen, dass es entweder "zu viele Einsen" oder
"Zu viele Nullen" in den frühen Stadien der Sequenz; wenn mode = 1, große Werte von dieser Statistik zeigen
dass es entweder "zu viele Einsen" oder "zu viele Nullen" in den späten Stadien. Kleine Werte der Statistik
würde bedeuten, dass Einsen und Nullen zu gleichmäßig miteinander vermischt.

2.13.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 100 Bit bestehen (dh, n =
100).

2.13.8 beispiel
(Eingang) e
= 11001001000011111101101010100010001000010110100011
00001000110100110001001100011001100010100010111000
(Eingang) n = 100
(Eingang) mode = 0 (vorwärts) | | mode = 1 (Rückseite)
(Verarbeitungs-) z = 1,6 (vorne) | | z = 1,9 (rückwärts)
(Output) P-Wert = 0,219194 (vorwärts) | | P-Wert = 0,114866 (reverse)
(Schluss) Da P-Wert> 0,01, akzeptieren die Sequenz als zufällig.

2,14 Zufällige Ausflüge Testen
2.14.1 Test-Purpose
Der Fokus dieses Tests ist die Anzahl der Zyklen mit genau K Besucher in eine kumulative Summe random walk.
Die kumulative Summe Irrfahrt von Teilsummen abgeleitet nach der (0,1)-Sequenz zu übertragen ist
die entsprechende (-1, +1)-Sequenz. Ein Zyklus von einem Random Walk besteht aus einer Abfolge von Schritten der Längeneinheit
genommen zufällig, dass am Anfang und Rückkehr zum Ursprung. Der Zweck dieses Tests ist es festzustellen, ob die
Anzahl der Besuche auf einem bestimmten Zustand innerhalb eines Zyklus abweicht, was man für eine zufällige erwarten
Sequenz. Dieser Test ist eigentlich eine Serie von acht Tests (und Schlussfolgerungen), eine Test-und Abschluss für jeden
das heißt: -4, -3, -2, -1 und +1, +2, +3, +4.

2.14.2 Function Call
RandomExcursions (n), wobei:
n Die Länge des Bit-String.
Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2-33

Analyse

2.14.3
Teststatistik und Reference Vertrieb
.2 (Obs):
Für einen gegebenen Zustand x, ein Maß dafür, wie gut die beobachtete Anzahl von Staatsbesuchen in einem
Zyklus mit dem erwarteten Anzahl von Staatsbesuchen innerhalb eines Zyklus, unter der Annahme einer
Zufälligkeit.

Die Referenz-Verteilung für die Teststatistik ist der 0,2-Distribution.

2.14.4
Test Beschreibung
(1)
Form einer normalisierten (-1, +1)-Sequenz X: Die Nullen und Einsen der Eingabesequenz (e) werden geändert
auf Werte von -1 und +1 über Xi = 2ei - 1.
Zum Beispiel, wenn e
= 0110110101, dann n = 10 und X = -1, 1, 1, -1, 1, 1, -1, 1, -1, 1.

(2)
Berechnen Sie die Partialsummen Si sukzessive größer Teilfolgen jeweils beginnend mit X1. Form der
Menge S = {Si}.
S1 = X1
S2 = X1 + X2
S3 = X1 + X2 + X3
.
.
Sk = X1 + X2 + X3 + ... + Xk
.
.
Sn = X1 + X2 + X3 + ... + Xk + ... + Xn

Für das Beispiel in diesem Abschnitt

S1 =- 1 S6 = 2
S2 = 0 S7 = 1
S3 = 1 S8 = 2
S4 = 0 S9 = 1
S5 = 1 S10 = 2

Die Menge S = {-1, 0, 1, 0, 1, 2, 1, 2, 1, 2}.

(3)
Form einer neuen Sequenz S 'durch das Anbringen Nullen vor und nach dem Satz S. Das heißt, S' = 0, S1, S2, ..., sn,
0 ist.
Für das Beispiel in diesem Abschnitt, S '= 0, -1, 0, 1, 0, 1, 2,1, 2, 1, 2, 0. Die daraus resultierende Irrfahrt
unten gezeigt.

Beispiel Random Walk (S ')

2-34

Analyse

(4)
Sei J = die Gesamtzahl der Nulldurchgänge in S ', wo ein Nulldurchgang einen Wert von Null in S ist '', dass
tritt nach dem Start gleich Null. J ist auch die Anzahl der Zyklen in S ', wo ein Zyklus von S'
ist ein
Teilfolge von S'consisting des Auftretens von Null, durch die keine Null-Werte gefolgt, und endet
mit einem anderen gleich Null. Die Endung Null in einem Zyklus kann zu Beginn Null in einem anderen Zyklus. Sterben
Anzahl der Zyklen in S 'ist die Anzahl der Nulldurchgänge. Wenn J <500, brechen Sie die test6.
Für das Beispiel in diesem Abschnitt, wenn S '= {0, -1, 0 1, 0, 1,2, 1, 2, 1, 2, 0}, dann J = 3 (es gibt Nullen
in den Positionen 3, 5 und 12 der S '). Die Nulldurchgänge sind leicht in den oben genannten Grundstück beobachtet. Da J =
3 gibt es 3 Zyklen, bestehend aus {0, -1, 0}, {0, 1, 0} und {0, 1, 2, 1, 2, 1, 2, 0}.

(5)
Für jeden Zyklus und für jeden Nicht-Null-Status-Wert x mit Werten -4 =
x =
-1 Und 1 =
x =
4,

Berechnen Sie die Frequenz jedes x in jedem Zyklus.
Für das Beispiel in diesem Abschnitt, der in Schritt 3, hat den ersten Zyklus ein Vorkommen von -1, die zweite
Zyklus hat ein Vorkommen von 1, und der dritte Zyklus hat drei Vorkommen von jeweils 1 und 2. Dies kann
visualisiert werden anhand der folgenden Tabelle.

Zustand
x
Cycles
Zyklus 1
(0, -1, 0)
Zyklus 2
(0, 1, 0)
Zyklus 3
(0,1,2,1,2,1,2,0)
-4 0 0 0
-3 0 0 0
-2 0 0 0
-1 1 0 0
1 0 1 3
2 0 0 3
3 0 0 0
4 0 0 0

(6)
Für jede der acht Staaten von x, zu berechnen. K (x) = die Gesamtzahl der Zyklen, in welchem Zustand x
tritt genau k-mal unter allen Zyklen, für k = 0, 1, ..., 5 (für k = 5, alle Frequenzen = 5 sind
abgelegt in .5 (x)). Beachten Sie, dass
!
=
50kkJ) "
.

Für das Beispiel in diesem Abschnitt

- .0 (-1) = 2 (die -1 Zustand tritt genau 0 mal in zwei Zyklen),
  .1 (-1) = 1 (die -1 Zustand tritt nur einmal in 1 Zyklus), und
  .2 (-1) = .3 (-1) = .4 (-1) = .5 (-1) = 0 (die -1 Zustand tritt genau {2, 3, 4, = 5} Zeiten in 0 Zyklen).

- .0 (1) = 1 (1-Zustand tritt genau 0 mal in 1 Zyklus),
  .1 (1) = 1 (1-Zustand tritt nur einmal in 1 Zyklus),
  6 J-mal das Minimum der Wahrscheinlichkeiten in der Tabelle in Abschnitt 3.14

gefunden werden muss = sein
5, um zu erfüllen die empirische Regel für
Chi-Quadrat-Berechnungen.

2-35

Analyse

.3 (1) = 1 (1-Zustand tritt genau dreimal in 1 Zyklus), und
.2 (1) = .4 (1) = .5 (1) = 0 (1-Zustand tritt genau {2, 4, = 5} mal in 0 Zyklen).

•
.0 (2) = 2 (die 2 Zustand tritt genau 0 mal in 2 Zyklen),
.3 (2) = 1 (die 2 Zustand tritt genau dreimal in 1 Zyklus), und
.1 (2) = .2 (2) = .4 (2) = .5 (2) = 0 (1-Zustand tritt genau {1, 2, 4, = 5} mal in 0 Zyklen).

•
.0 (-4) = 3 (von -4 Zustand tritt genau 0 mal in 3 Zyklen) und
.1 (-4) = .2 (-4) = .3 (-4) = .4 (-4) = .5 (-4) = 0 (der -4 Zustand tritt genau {1, 2, 3, 4, = 5} mal in 0 Zyklen).

Und so weiter ....

Dies kann gezeigt werden, anhand der folgenden Tabelle werden:

Staatliche x
Anzahl der Zyklen
0 1 2 3 4 5
-4 3 0 0 0 0 0
-3 3 0 0 0 0 0
-2 3 0 0 0 0 0
-1 2 1 0 0 0 0
1 1 1 0 1 0 0
2 2 0 0 1 0 0
3 3 0 0 0 0 0
4 3 0 0 0 0 0

(7) Für jede der acht Staaten von x, berechnen die Teststatistik
obs (
%,
wo pk (x) ist die Wahrscheinlichkeit, dass der Staat x k-mal auftritt, in einer zufälligen Verteilung (siehe Abschnitt
3.14 für eine Tabelle mit pk-Werte). Die Werte für pk (x) und ihre Methode der Berechnung sind in sofern

Abschnitt 3.14. Beachten Sie, dass acht 0,2 Statistiken produziert werden (dh für x = -4, -3, -2, -1, 1, 2, 3, 4).
Zum Beispiel in diesem Abschnitt, wenn x = 1, = "
= 4.333033

(8) Für jeden Staat der x, berechnen P-Wert = igamc (2.5, obs (2!. Eight P-Werte werden
produziert.

Für das Beispiel, wenn x = 1, P-Wert = igamc

$%
&
0,502529.

2.14.5 Entscheidung Rule (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

Analyse

2.14.6 Zusammenfassung und Interpretation der Ergebnisse
Da die P-Wert in Schritt 8 des Abschnitts 2.14.4 gewonnen wird =
0,01 (p-Wert = 0,502529), ist der Abschluss
dass die Reihenfolge ist zufällig.

Beachten Sie, dass, wenn .2 (obs) waren zu groß, dann ist die Folge wäre eine Abweichung von der angezeigten haben
theoretische Verteilung für einen bestimmten Zustand in allen Zyklen.

2.14.7 Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 1.000.000 Bits bestehen (dh, n =
106).

2.14.8 beispiel
(Eingang) e
= "Die binäre Ausbau von E bis zu 1.000.000 bits"
(Eingang) n = 1000000 = 106
(Verarbeitung) J = 1490

State = x 0,2 P-Wert Fazit
-4 3,835698 0,573306 Zufällige
-3 7,318707 0,197996 Zufällige
-2 7,861927 0,164011 Zufällige
-1 15,692617 0,007779 Non-random

+1 2.485906 0.778616 Zufällige
+2 5.429381 0.365752 Zufällige
+3 2.404171 0.790853 Zufällige
+4 2.393928 0.792378 Zufällige

(Schluss)
Für sieben von den Staaten der x, ist der P-Wert =
0,01, und die Schlussfolgerung wäre, dass die
Sequenz wurde zufällig. Doch für einen Zustand x (x = -1), die P-Wert <0,01 ist, so dass die
Schlussfolgerung wäre, dass die Reihenfolge nicht zufällig ist. Wenn Widersprüche auftreten, weitere
Sequenzen sollten untersucht werden, um festzustellen, ob dieses Verhalten typisch für die ist
Generator.

2,15 Zufällige Ausflüge Variant-Test
2.15.1 Test-Purpose
Der Fokus dieses Tests ist die Gesamtzahl der Zeiten, die einen bestimmten Zustand besucht wird (dh, tritt) in einem
kumulative Summe random walk. Der Zweck dieses Tests ist es, Abweichungen von der erwarteten Anzahl erkennen
der Besuche in verschiedenen Staaten in die Irrfahrt. Dieser Test ist eigentlich eine Reihe von achtzehn Prüfungen (und
Schlussfolgerungen), eine Test-und Schluss für jeden der Staaten: -9, -8, ..., -1 und +1, +2, ..., +9.

2.15.2 Function Call
RandomExcursionsVariant (n), wobei:

n Die Länge der Bitfolge, erhältlich als ein Parameter in der Funktion aufzurufen.

Analyse

Zusätzlicher Eingang von der Funktion verwendet, sondern versorgt die Testing-Code:

e
Die Folge von Bits als durch die RNG oder PRNG getestet generiert, das existiert als globale Struktur zum Zeitpunkt der Aufruf der Funktion, e
= E1, E2, ..., en.

2.15.3
Teststatistik und Reference Vertrieb
.:

Für einen gegebenen Zustand x, die Gesamtzahl der Zeit, dass die gegebenen Zustand ist während der besuchten
gesamte Irrfahrt wie in Schritt 4 des Abschnitts 2.15.4 bestimmt.

Die Referenz-Verteilung für die Teststatistik ist die Hälfte normal (für große n). (Hinweis: Wenn.
ist als verteilte
normal, dann |.. | ist die Hälfte normalverteilt) Wenn die Reihenfolge ist zufällig, dann ist die Teststatistik wird
etwa 0. Wenn es zu viele Einsen oder zu viele Nullen sind, dann ist die Teststatistik wird groß sein.

2.15.4
Test Beschreibung
(1)
Form der normierten (-1, +1)-Sequenz X, in der die Nullen und Einsen der Eingabesequenz (e)
1 -. sind auf Werte von -1 und +1 via X = X1, X2, ..., Xn, wo Xi = 2ei umgewandelt
Zum Beispiel, wenn e
= 0110110101, dann n = 10 und X = -1, 1, 1, -1, 1, 1, -1, 1, -1, 1.

(2)
Compute Partialsummen Si sukzessive größer Teilfolgen jeweils beginnend mit x1.
Bilden die Menge S
= {Si}.
S1 = X1
S2 = X1 + X2
S3 = X1 + X2 + X3
.
.
Sk = X1 + X2 + X3 +. . . + Xk
.
.
Sn = X1 + X2 + X3 +. . . + Xk +. . . + Xn

Für das Beispiel in diesem Abschnitt

S1 =- 1 S6 = 2
S2 = 0 S7 = 1
S3 = 1 S8 = 2
S4 = 0 S9 = 1
S5 = 1 S10 = 2

Die Menge S = {-1, 0, 1, 0, 1, 2, 1, 2, 1, 2}.

(3)

Form einer neuen Sequenz S 'durch das Anbringen Nullen vor und nach dem Satz S. Das heißt, S' = 0, S1, S2, ..., sn,
0 ist.
2-38

Analyse

Für das Beispiel, S '= 0, -1, 0, 1, 0, 1,2, 1, 2,1, 2, 0. Die daraus resultierende Irrfahrt ist unten dargestellt.

Beispiel Random Walk (S ')
(4)
Für jedes der achtzehn Nicht-Null-Zustände x, zu berechnen. (X) = die Gesamtzahl der Zeiten, die staatliche x
aufgetreten allen J-Zyklen.
Für das Beispiel in diesem Abschnitt. (-1) = 1,. (1) = 4,. (2) = 3, und alle anderen. (X) = 0.

(5) Für jede. (X), berechnet P-Wert = erfc $ $ $%
2) (
(
. Achtzehn P-Werte berechnet werden. Sehen
Abschnitt 5.5.3 für die Definition von erfc.

Für das Beispiel in diesem Abschnitt, wenn x = 1, P-Wert = erfc
$ $%
0,683091.

2.15.5
Entscheidungsregel (auf dem 1% Level)
Wenn der berechnete P-Wert <0,01, dann folgern, dass die Reihenfolge nicht zufällig ist. Andernfalls schließen
dass die Reihenfolge ist zufällig.

2.15.6
Fazit und Interpretation der Ergebnisse
Da die P-Wert in Schritt 7 des Abschnitts 2.15.4 gewonnen wird =

0,01 für den Zustand x = 1 (P-Wert = 0,683091),
Die Schlussfolgerung ist, dass die Reihenfolge zufällig ist.

2.15.7
Input Size Empfehlung
Es wird empfohlen, dass jede Sequenz zu prüfenden von mindestens 1.000.000 Bits bestehen (dh, n =
106).

2.15.8
Beispiel
(Eingang) e
= "Die binäre Ausbau von E bis zu 1.000.000 bits"
(Eingang) n = 1000000 = 106

(Verarbeitung) J = 1490

Analyse

State (x) Zählt P-Wert Fazit
-9 1450 0,858946 Zufällige
-8 1435 0,794755 Zufällige
-7 1380 0,576249 Zufällige
-6 1366 0,493417 Zufällige
-5 1412 0,633873 Zufällige
-4 1475 0,917283 Zufällige
-3 1480 0,934708 Zufällige
-2 1.468 0,816012 Zufällige
-1 1502 0,826009 Zufällige
1 1409 0,137861 Zufällige
2 1369 0,200642 Zufällige
3 1396 0,441254 Zufällige
4 1479 0,939291 Zufällige
5 1599 0,505683 Zufällige
6 1628 0,445935 Zufällige
7 1619 0,512207 Zufällige
8 1620 0,538635 Zufällige
9 1610 0,593930 Zufällige

(Schluss) Da die P-Wert =
0,01 für jede der achtzehn Staaten von x, akzeptieren die Sequenz als zufällig.

Analyse

3. Technische Beschreibung des Tests
Dieser Abschnitt enthält die mathematischen Grundlagen für die Tests in der NIST Testsuite. Jeder Unterabschnitt
entspricht dem entsprechenden Unterabschnitt in Abschnitt 2. Die entsprechenden Referenzen für jede Teilstrecke werden
sofern am Ende dieses Absatzes.

3,1 Frequency (Monobits) Test
Die einfachste Test ist, dass der Null-Hypothese: in eine Folge von unabhängigen, identisch verteilten
Bernoulli Zufallsvariablen (X-oder e ist, wo X = 2e-1, und so Sn = X1 + ... + Xn = 2 (e1 + ... + en) - n), die
Wahrscheinlichkeit von Einsen ist 1 / 2. Mit dem klassischen De Moivre-Laplace, für eine ausreichend große Anzahl von

Versuche, die Verteilung der binomial Summe von normalisierten
n, ist eng mit einem Standard angenähert

Normalverteilung. Dieser Test nutzt die Annäherung an die Nähe der Bruchteil der Beurteilung
1er-bis 1 / 2. Alle nachfolgenden Tests basieren auf dem Passieren dieser ersten grundlegenden Test konditioniert.

Der Test wird von dem bekannten Grenzwert Zentraler Grenzwertsatz für die Irrfahrt abgeleitet, Sn = X1 + ... +
Xn. Nach dem zentralen Grenzwertsatz,

limn "#
P (
Snn
$ Z) = (1)
Dieses klassische Ergebnis dient als Grundlage der einfachste Test auf Zufälligkeit. Es impliziert, dass für positive z,

P (
Snn
"Z) = 2 # Nach dem Test auf die Statistik s Basis =
Sn /
n, bewerten den beobachteten Wert
s (obs)
=
X1 + K + Xn /
n, und berechnen Sie dann die entsprechende P-Wert, der

21 "# (s (obs))[]=. Hier wird erfc der (ergänzenden) Fehlerfunktion
erfc (z) =
2
"
$%
.
Referenzen für Test

[1]
Kai Lai Chung, Elementare Wahrscheinlichkeitstheorie mit Stochastic Processes. New York: Springer-
Verlag, 1979 (bes. S. 210-217).
[2]
Jim Pitman, Probability. New York: Springer-Verlag, 1993 (insbesondere S. 93-108).
3.2 Frequenzschutz-Test innerhalb eines Blocks
Der Test zielt auf lokale Abweichungen von der idealen 50% Frequenz von 1 ist zu erkennen durch die Zersetzung des Tests
Sequenz in einer Reihe von überlappenden Teilsequenzen und Anwendung eines Chi-Quadrat-Test für eine
homogene Spiel der empirischen Frequenzen an die ideal 1 / 2. Kleine P-Werte zeigen große Abweichungen
von der gleiche Anteil von Einsen und Nullen in mindestens einer der Teilstrings. Die Zeichenfolge von 0 und 1 (bzw.
entspricht -1 's und 1) ist in eine Anzahl disjunkter Teilstrings unterteilt. Für jeden Teilstring, der
Anteil der Einsen berechnet. Ein Chi-Quadrat-Statistik vergleicht diese substring Proportionen, die ideale

Analyse

1 / 2. Die Statistik ist es, einen Chi-Quadrat-Verteilung mit den Freiheitsgraden, die der Anzahl nach
von Teilstrings.

Die Parameter dieses Tests sind M und N, so dass n = MN, dh die ursprüngliche Zeichenfolge in N partitioniert ist
Teilstrings, die jeweils mit einer Länge M. Für jede dieser Zeichenketten ist die Wahrscheinlichkeit von Einsen durch die geschätzte
"Iobserved relative Häufigkeit von 1 ist,, i = 1, ..., N. Die Summe

"2 (obs) = 4mi = +
unter der Zufälligkeit Hypothese hat die .2-Verteilung mit N Freiheitsgraden. Das berichtet pValue
ist

e "u / 2
2 (obs)
$% UN / 2 & (N / 2) 2 N / 2 =
.
Referenzen für Test

[1]
Nick Maclaren "Cryptographic Pseudo-Zufallszahlen in Simulation," Cambridge Sicherheit
Workshop zum Thema Fast Software Encryption. Dezember 1993. Cambridge, GB: R. Anderson, S. 185-190.
[2]
Donald E. Knuth, The Art of Computer Programming. Vol. 2: Seminumerical Algorithms. 3. Aufl..
Reading, Mass: Addison-Wesley, 1998 (insbesondere S. 42-47).
[3]
Milton Abramowitz und Irene Stegun, Handbook of Mathematical Functions: NBS Applied
Mathematics Series 55. Washington, DC: US Government Printing Office, 1967.
3.3 läuft Testen
Diese Variante des klassischen parametrischen Test befasst sich mit dem "es" als Teilstrings von aufeinander folgenden 1 ist definiert und
Folge 0 ist, und prüft, ob die Oszillation zwischen diesen homogenen Teilstrings zu schnell oder ist
zu langsam.

Die spezifischen Test verwendet hier auf die Verteilung der Gesamtzahl der Durchläufe, Vn basiert. Für die feste

Anteil
"= # J / nj $
(Die durch die Frequenz-Test von Abschnitt 3.1 muss festgestellt wurden, werden
nahe 0,5:
"#
12
$
2n).
limn "#
P (
Vn $ 2n% (2n% (1 (2)
Zur Beurteilung Vn, für k = 1, ..., n-1 zu definieren,

r (k) = 0
WENN

"K =" k +1
und

r (k) = 1
WENN
"K #" k +1. Dann

Vn = r (k) +1 n "1 #
. Der P-Wert gemeldet wird

erfc (
Vn (obs) "2n22n #).
3-2

Analyse

Große Werte von

Vn (obs)
zeigen Schwingung in der Folge von e ist, die zu schnell ist, kleine Werte zeigen

Schwingung, die zu langsam ist.

Referenzen für Test

[1]
Jean D. Gibbons, Nichtparametrische Statistische Inferenz, 2. Aufl.. New York: Marcel Dekker, 1985
(Insbesondere S. 50-58).
[2]
Anant P. Godbole und Stavros G. Papastavridis, (ed), Runs und Muster in Wahrscheinlichkeit: Selected
Papiere. Dordrecht: Kluwer Academic, 1994.
3.4 Test für den Longest Run of Ones in einem Block
Die Länge der längsten Folge Teilfolge (run) von Einsen ist ein weiteres Merkmal, das verwendet werden kann
für die Prüfung Zufälligkeit. Ein String der Länge n, so dass n = MN, muss in partitioniert werden
"JN Teilstrings,
jede Länge M. Für den Test auf die Länge der längsten Piste von Einsen auf

innerhalb der j-ten substring
Größe M, K + 1 Klassen gewählt werden (je nach M). Für jede dieser Teilstrings, wertet man die
, Dh die berechneten Werte der längste Abfahrt von Einsen

Programmleitung
"0", 1, K, "k (" 0 + innerhalb jeder dieser Teilstrings Zugehörigkeit zu einem der K + 1

gewählt Klassen). Wenn es r Einsen und M

r Nullen in der m-Bit-Block, dann ist die bedingte Wahrscheinlichkeit, dass die längste Folge von Nullen
"
in diesem Block
ist kleiner oder gleich m hat die folgende Form mit
U = min (M "r + (siehe [1]):
P ("# m | r) =
$
%
&
,
so dass

P ("# m) =
Herr
$
%
&
r = 0M
*.
(3)

Die theoretischen Wahrscheinlichkeiten

"0", 1, K ", K
Diese Klassen sind aus [3] bestimmt.

Die empirischen Frequenzen

"I, i = 0, K, K
werden von den verwachsenen
"2-Statistik

"2 =
I $ N%% ii = 0K
&,
, die unter die Zufälligkeit Hypothese, hat eine ungefähre
"2-Verteilung mit K Freiheitsgraden.
Die ausgewiesenen P-Wert ist

Analyse

e "u/2uK/2 # 2 (obs)
$%
& K / 2 () 2K / 2 =,
mit P (a, x) bezeichnet die unvollständige Gamma-Funktion, wie in Abschnitt 3.2 angegeben.

Die folgende Tabelle enthält ausgewählte Werte von K und M mit den entsprechenden Wahrscheinlichkeiten erhalten
aus [3]. Fälle K = 3, M = 8; K = 5, M = 128, und K = 6, M = 10000 sind derzeit in der Test-Suite eingebettet
Code.

K = 3, M = 8

Klassen Wahrscheinlichkeiten
{1} .= p0
= 0,2148
{2} .= p1 = 0,3672
{3} .= p2 = 0,2305
{4} .= p3 = 0,1875

K = 5, M = 128

Klassen Wahrscheinlichkeiten
{4} .= p0
= 0,1174
{5} .= p1 = 0,2430
{6} .= p2 = 0,2493
{7} .= p3 = 0,1752
{8} .= p4 = 0,1027
{9} .= p5 = 0,1124

K = 5, M = 512

Klassen Wahrscheinlichkeiten
{6} .= p0
= 0,1170
{7} .= p1 = 0,2460
{8} .= p2 = 0,2523
{9} .= p3 = 0,1755
{10} .= p4 = 0,1027
{11} .= p5 = 0,1124

K = 5, M = 1000

Klassen Wahrscheinlichkeiten
{7} .= p0
= 0,1307
{8} .= p1 = 0,2437
{.= 9} p2 = 0,2452
{10} .= p3 = 0,1714
{11} .= p4 = 0,1002
{12} .= p5 = 0,1088

3-4

K = 6, M = 10000

Klassen Wahrscheinlichkeiten
{10} .= p0
= 0.0882
{11} .= p1 = 0,2092
{.= 12} p2 = 0,2483
{13} .= p3 = 0,1933
{14} .= p4 = 0,1208
{15} .= p5 = 0.0675
{16} .= p6 = 0,0727

Große Werte von

"2
zeigen, dass die Sequenz-Clustern von Einsen hat, die Erzeugung von "random
"N"-Sequenzen
von Menschen neigt dazu, auf kleine Werte von Blei
(Siehe [3, S. 55]).
Referenzen für Test

[1]
FN David und DE Barton, Kombinatorische Chance. New York: Hafner Publishing Co.,
1962, S.
230.
[2]
Anant P. Godbole und Stavros G. Papastavridis (ed), Runs und Patterns in Probability:
Selected
Papers. Dordrecht: Kluwer Academic, 1994.

[3]
Pal Revesz, Random Walk in Random und Non-Random-Umgebungen. Singapore: World
Scientific, 1990.
3,5 Binary Matrix Rank Test
Ein weiterer Ansatz zur Untersuchung Zufall ist es, für die lineare Abhängigkeit zwischen fester Länge zu überprüfen
Teilstrings der ursprünglichen Sequenz: Bau-Matrizen von aufeinanderfolgenden Nullen und Einsen aus der Sequenz,
und für die lineare Abhängigkeit zwischen den Zeilen oder Spalten der konstruierten Matrizen zu überprüfen. Die Abweichung der
den Rang-oder Rang-Mangel-der Matrizen aus einer theoretisch erwarteten Wert gibt die Statistik der
Interesse.

Dieser Test ist eine Spezifikation, einer der Tests aus dem DIEHARD [1] Testbatterie. Es basiert
auf das Ergebnis der Kovalenko [2] und auch von Marsaglia und Tsay formuliert [3]. Das Ergebnis besagt, dass die
Rang R der MxQ zufälligen binären Matrix nimmt die Werte r = 0, 1, 2, ..., m, wo m = min (M, Q) mit
Wahrscheinlichkeiten

pr = 2r (Q + M "r)" MQ1.
Die Wahrscheinlichkeit, dass Werte in die Test-Suite-Code für M = Q = 32 fixiert. Die Zahl M ist dann ein Parameter
dieses Tests, so dass im Idealfall n = M2N, wobei N die neue "sample size." In der Praxis sind Werte für M und N
so dass die abgelegten Teil des Strings, n-NM2, ziemlich klein gewählt wird.

Der Grund für diese Wahl ist, dass

Analyse

pM "1 #
12j
$
% &
"
() J = 1
* +
,
pM "1 # # 2.00 0,5776 ...
,

pM "2 #
4pM9
0,1284 ...
und alle anderen Wahrscheinlichkeiten sind sehr klein (=
0,005), wenn M =
10.

Für die N quadratische Matrizen erhalten, ihre Reihen

Rl, l = 1, K, N
ausgewertet werden, und die Frequenzen

FM, FM "1
Und

N "FM" FM "1
der Werte M, M-1 und der Ränge von nicht mehr als M-2 bestimmt sind:

FM = # Rl = M {}
,
FM "1 = # Rl = M.
Zur Anwendung der
"2-Test, mit dem klassischen Statistik

"2 =
FM # 0.2888N (), die unter die Null (Zufälligkeit) Hypothese, hat eine ungefähre
"2
exp "# 2 (obs) / 2 {}
-Verteilung mit 2 Grad
Freiheit. Die ausgewiesenen P-Wert ist.

Interpretation dieses Tests: große Werte von

"2 (obs)
zeigen, dass die Abweichung der Verteilung von Rang
dass entsprechend einer zufälligen Reihenfolge ist wichtig. Zum Beispiel produziert
Pseudo-Random-Matrizen
durch ein Schieberegister-Generator, der von weniger als M aufeinanderfolgende
Vektoren gebildet systematisch Rang

Rl "M,
während für wirklich zufällige Daten, sollte der Anteil solcher Ereignisse nur etwa 0,29
sein.

Referenzen für Test

[1]

George Marsaglia, DIEHARD: eine Batterie von der Zufälligkeit.
http://www.stat.fsu.edu/pub/diehard/.
[2]
IN Kovalenko (1972), "Distribution der linearen Rang einer zufälligen Matrix" Theory of Probability and its Applications. 17, S. 342-346.
[3]
G. Marsaglia und LH Tsay (1985), "Matrizen und die Struktur der Zufallszahlenfolgen" Lineare Algebra und ihre Anwendungen. Vol. 67, S. 147-156.
3,6 Diskrete Fourier-Transformation (Specral) Test
Die hier beschriebenen Test basiert auf der diskreten Fourier-Transformation. Es ist ein Mitglied einer Klasse von Verfahren
bekannt als spektrale Methoden. Die Fourier-Test erkennt periodische Funktionen in der Bit-Serie, würde bedeuten,
eine Abweichung von der Annahme des Zufalls.

Lassen xk die k-te Bit, wobei k = 1, ..., n. Angenommen, die Bits sind -1 und +1 kodiert. Lassen

Analyse

fj = xkexp2k = 1N
$
,
WO
exp2 "IKJ / n () = cos2and
i "# 1. Wegen der
Symmetrie der realen hin zu komplexen-Wert zu verwandeln, nur die Werte von 0 bis (n / 2 -1) werden berücksichtigt. Lassen
modj der Modul der komplexen Zahl fj werden. Unter der Annahme der Zufälligkeit der Reihe xi, ein
Konfidenzintervall kann auf die Werte der platziert werden
h = log10.05
"
#
$
%
&
"Genauer gesagt, 95 Prozent der Werte von

modj sollte weniger als
. Ein P-Wert für diese Schwelle Basis kommt aus dem binomischen

Verteilung. Lassen N1 die Anzahl der Spitzen weniger als h. werden Nur die ersten n / 2 Peaks werden berücksichtigt. Lassen N0 =

0,95 N / 2 und
d = N1 "N0 () / Der P-Wert ist
21 "# d ()()= erfcd2
wo f (x) ist die kumulative Wahrscheinlichkeit in Abhängigkeit von der Standard-Normalverteilung.

Andere P-Werte in der Serie fj oder modj basiert, die empfindlich auf Abweichungen sind aus Zufall sind
Möglich ist. Allerdings kommt der primäre Wert der Transformation von einer Handlung der Serie modj. Im
nebenstehende Abbildung zeigt die obere Plot der Serie modj für 4096 Bits von einer zufriedenstellenden generiert
Generator. Die Linie durch die Handlung ist das 95% Grenze. Der P-Wert für diese Serie ist
0,8077. Die untere Kurve zeigt ein entsprechendes Grundstück für einen Generator, der Bits, die produziert
statistisch abhängig in ein periodisches Muster. In der unteren Grundstück, deutlich größer als 5% der
Größen sind darüber hinaus das Vertrauen Grenze. Darüber hinaus gibt es eine klare Struktur in den Größen
das ist nicht im oberen Plot. Der P-Wert für diese Serie ist 0,0001.

Referenzen für Test

[1]
RN Bracewell, Die Fourier Transform and Its Applications. New York: McGraw-Hill, 1986.
[2]
W. Killman, J. Schüth, W. Thumser, und I. Uludag, Ein Hinweis bezüglich der DFT Test in NIST
Special Publication 800-22, T-Systems, Systems Integration, Juli 2004.
[3]
S. Kim, K, Umeno und A. Hasegawa, Korrekturen des NIST Statistische Test Suite für Randomness, Kryptologie ePrint Archive, Report 2004/018, 2004.

Analyse

Analyse

3,7 nicht-überlappenden Template Matching-Test
Dieser Test weist Sequenzen, zu viele oder zu wenige Instanzen eines gegebenen aperiodische Muster.

Lassen
B = "10, K," m0 ()
werden ein gegebenes Wort (Vorlage oder Muster, dh, eine feste Abfolge von Nullen und Einsen) der

Länge m. Dieses Muster ist so zu wählen, als ob es einen Parameter des Tests waren. Wir betrachten einen Test auf der Grundlage
Muster für feste Länge m. Eine Tabelle mit ausgewählten aperiodische Worte aus solchen Mustern für m = 2, ..., 8 ist
sofern am Ende dieses Abschnitts.

Die Menge der Perioden von B

"= J, 1 # j # m,
Eine wichtige Rolle spielt. Zum Beispiel, wenn B entspricht einer Auflage von m diejenigen,
"= 1, K, m # 1 {}
. Für sterben

B oben
"= #
Und B ist eine aperiodische Muster (dh es kann nicht geschrieben werden als

CCKC "C
für ein Muster C
kürzer als B mit C 'bezeichnet das Präfix C). In dieser Situation sind die Vorkommen von B in der Zeichenfolge nicht überlappende.

Im Allgemeinen lassen

W = W (m, M)
die Anzahl der Vorkommen der gegebenen Muster B in der Zeichenkette. Note
dass die Statistik W ist auch für Muster B mit definierten
"= #
. Der beste Weg, um W zu berechnen ist als die Summe,

W = I "i + k # 1 = i = 1 N # m +1
$
.
Die Zufallsvariablen

I "i + k # 1 =" k0, k = 1, m-abhängig, so dass der zentrale Grenzwertsatz

gilt für W. Der Mittelwert und die Varianz der approximierenden Normalverteilung folgende Form haben,

μ =
n "m +12 m,
"2 = n12m #
2m $
% &
.
Für die Test-Suite-Code, M
Wj = Wj (m, M)
und N sind so gewählt, dass n = MN und N = 8. Partitionieren Sie die ursprüngliche Zeichenfolge in N
Blöcke der Länge M. Let

werden die Anzahl der Vorkommen des Musters B in den Block j, für

j = 1, ..., N.

. Dann wird für große M hat Wj einer Normalverteilung mit Mittelwert μ und

Varianz s2, so dass die Statistik

Lassen
μ = EWJ = (M "," 2 (obs) =
Wjμ () 1N
%
(4)
3-9

Analyse

-Verteilung mit N Freiheitsgraden. Melden Sie den P-Wert als

hat eine ungefähre
"2
1 "PN2,
2 (obs)
2
$
%
&

.
Der Test kann als Ablehnung interpretiert werden Sequenzen aufweisen unregelmäßige Vorkommen eines bestimmten nicht-periodische
Muster.

Referenzen für Test

[1]
AD Barbour, L. Holst und S. Janson, Poisson Approximation (1992). Oxford: Clarendon Press
(Insbesondere Abschnitt 8.4 und Abschnitt 10.4).
Aperiodische Vorlagen für kleine Werte von

2 "m" 5
m = 2 m = 3 m = 4 m = 5
0 1 0 0 1 0 0 0 1 0 0 0 0 1
1 0 0 1 1
1 0 0
0 0 1 1
0 1 1 1
0 0 0 1 1
0 0 1 0 1
1 1 0 1 0 0 0
1 1 0 0
1 1 1 0
0 1 0 1 1
0 0 1 1 1
0 1 1 1 1
1 1 1 0 0
1 1 0 1 0
1 0 1 0 0
1 1 0 0 0
1 0 0 0 0
1 1 1 1 0

Analyse

Aperiodische Vorlagen für kleine Werte von

6 "m" 8
m = 6 m = 7 m = 8
0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1
0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1
0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1

0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1
0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1
0 0 1 1 0 1 0 0 0 1 0 1 1 0 0 0 0 1 0 1 1
0 0 1 1 1 1 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1
0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
0 1 0 1 1 1 0 0 1 0 0 1 1 0 0 0 1 0 0 1 1
0 1 1 1 1 1 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1
1 0 0 0 0 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1
1 0 1 0 0 0 0 0 1 1 0 1 1 0 0 0 1 1 0 0 1
1 0 1 1 0 0 0 0 1 1 1 0 1 0 0 0 1 1 0 1 1
1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 0 1
1 1 0 0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1
1 1 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1
1 1 1 0 0 0 0 1 0 1 0 1 1 0 0 1 0 0 1 0 1
1 1 1 0 1 0 0 1 0 1 1 1 1 0 0 1 0 0 1 1 1
1 1 1 1 0 0 0 1 1 0 1 1 1 0 0 1 0 1 0 1 1
1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 1 0 1 1 0 1
1 0 0 0 0 0 0 0 0 1 0 1 1 1 1
1 0 0 1 0 0 0 0 0 1 1 0 1 0 1
1 0 1 0 0 0 0 0 0 1 1 0 1 1 1
1 0 1 0 1 0 0 0 0 1 1 1 0 1 1
1 0 1 1 0 0 0 0 0 1 1 1 1 0 1
1 0 1 1 1 0 0 0 0 1 1 1 1 1 1
1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1
1 1 0 0 0 1 0 0 1 0 0 0 1 1 1
1 1 0 0 1 0 0 0 1 0 0 1 0 1 1
1 1 0 1 0 0 0 0 1 0 0 1 1 1 1
1 1 0 1 0 1 0 0 1 0 1 0 0 1 1
1 1 0 1 1 0 0 0 1 0 1 0 1 1 1
1 1 1 0 0 0 0 0 1 0 1 1 0 1 1
1 1 1 0 0 1 0 0 1 0 1 1 1 1 1
1 1 1 0 1 0 0 0 1 1 0 0 1 1 1
1 1 1 0 1 1 0 0 1 1 0 1 1 1 1
1 1 1 1 0 0 0 0 1 1 1 1 1 1 1
1 1 1 1 0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 1 0 0 1 0 0 1 0 0 0 0
1 1 1 1 1 1 0 1 0 0 1 1 0 0 0
1 0 1 0 0 0 0 0
1 0 1 0 0 1 0 0
1 0 1 0 1 0 0 0
1 0 1 0 1 1 0 0
1 0 1 1 0 0 0 0
1 0 1 1 0 1 0 0
1 0 1 1 1 0 0 0
1 0 1 1 1 1 0 0
1 1 0 0 0 0 0 0
1 1 0 0 0 0 1 0
1 1 0 0 0 1 0 0

1 1 0 0 1 0 0 0
1 1 0 0 1 0 1 0
1 1 0 1 0 0 0 0
1 1 0 1 0 0 1 0
1 1 0 1 0 1 0 0
1 1 0 1 1 0 0 0
1 1 0 1 1 0 1 0
1 1 0 1 1 1 0 0
1 1 1 0 0 0 0 0
1 1 1 0 0 0 1 0
1 1 1 0 0 1 0 0
1 1 1 0 0 1 1 0
1 1 1 0 1 0 0 0
1 1 1 0 1 0 1 0
1 1 1 0 1 1 0 0
1 1 1 1 0 0 0 0
1 1 1 1 0 0 1 0
1 1 1 1 0 1 0 0
1 1 1 1 0 1 1 0
1 1 1 1 1 0 0 0
1 1 1 1 1 0 1 0
1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 0

3-11

Analyse

3,8 Overlapping Template Matching-Test
Dieser Test weist Sequenzen, die zu viele oder zu wenige Vorkommen von m-Läufe von denen zu zeigen, aber kann
leicht modifiziert, um irreguläre Vorkommnisse eines beliebigen periodischen Muster B. erkennen

Zur Umsetzung dieses Tests werden die Parameter M und N so bestimmt, dass n = MN, dh, wird die ursprüngliche Zeichenfolge
partitioniert in N Blöcke mit jeweils der Länge M.

Lassen
~ W j = ~ W jm, n ()
die Anzahl der (sich möglicherweise überschneidenden) läuft von Einsen der Länge m in der j-ten Block.

Die asymptotische Verteilung von Wj ist die Verbindung Poisson-Verteilung (die sogenannte Polya-Aeppli Gesetz,
siehe [1]):

Eexpt ~ W j {} "exp
(5)
WENN
M "m +1 () 2" m # (t ist eine echte Variable).

Die entsprechenden Wahrscheinlichkeiten in Bezug auf die konfluenten hypergeometrischen Funktion ausgedrückt werden

"= 1F1. If U bezeichnet eine Zufallsvariable mit der Verbindung Poisson asymptotische Verteilung, dann für u =
1 mit
"= # / 2.
PU = u () =
e "#
2uul =*.
Zum Beispiel,

VE = 0 () = e "#
,
VE = 1 () =
"
2e # "
,
VE = 2 () =
"E #"
8
,
VE = 3 () =
"E #"
8
,
VE = 4 () =
"E #"
16
.
Genauer, aber auch komplizierter werden Ausdrücke in [3] gegeben. Diese Ausdrücke ergeben die
folgenden Wahrscheinlichkeiten:

p0 0,364091
p1 0,185659
p2 0,139381
p3 0,100571
p4 0,0704323
p5 0,139865

Analyse

Die Ergänzung der Verteilungsfunktion dieser Zufallsvariablen hat die Form

mit

Lu () = PU> u () = "l, u () =
12kk $
%
&
k = lu
*
Wählen Sie K + 1 Klassen oder Zellen für U, d. h.
U = 0 {}, U = 1 {},. Der theoretische
Wahrscheinlichkeiten
"0", 1, K ", K +1 dieser Zellen sind aus den obigen Formeln gefunden. Eine vernünftige Wahl
werden konnte
K = 5 "= 2, # = 1 ist.
Nach U1, ..., UN gefunden werden, bewerten die Frequenzen

"0", 1, K ", K
jeder Zelle,
"0 +" 1 + K + "K =

und berechnen den Wert des
"2-Statistik

"2 =
I $ N%% ii = 0K
&
.
Der Ausdruck für die P-Wert ist die gleiche wie die in Abschnitt 3.7 verwendet. Die Interpretation ist, dass für sehr
kleine P-Werte, zeigt die Sequenz unregelmäßige Vorkommen von m-Läufe von Einsen.

Referenzen für Test

[1]
O. Chrysaphinou und S. Papastavridis, "A Grenzwertsatz über die Anzahl der überlappenden
Der Schein eines Pattern in eine Folge von unabhängigen Studien.
"Wahrscheinlichkeitstheorie und Verwandte
Felder, vol. 79 (1988), S. 129-143.

[2]
NJ Johnson, S. Kotz und A. Kemp, diskrete Verteilungen. John Wiley, 2. Aufl.. New York,
1996 (insbesondere S. 378-379).
[3]
K. Hamano und T. Kaneko, die Korrektur der Overlapping Template Matching-Test inklusive
in NIST Randomness Test Suite, IEICE Transactions of Electronics, Communications and
Computer Sciences 2007, E90-A (9), pp 1788-1792.
3,9 Maurers "Universal Statistical" Test
Ueli Maurer von der Abteilung für Informatik an der Princeton University eingeführt diesem Test im Jahr 1992.
Maurer Teststatistik bezieht sich eng auf die pro-Bit-Entropie der Strom, der seinem Autor behauptet, ist "die
richtigen Qualität Maßnahme für einen geheimen Schlüssel Quelle in einem kryptographischen Anwendung. "Als solches ist der Test
behauptet, die eigentliche kryptographische Bedeutung des Mangels zu messen, weil sie "zu den laufenden Bezug
Zeit [an] Feindes optimale Schlüssel-Suche-Strategie ", oder die effektive Schlüssellänge einer Chiffre-System.

Der Test ist nicht dafür ausgelegt, eine sehr spezifische Muster oder Art der statistischen Fehler zu erkennen. Allerdings ist der Test
entwickelt ", um der Lage sein, eine der sehr allgemeinen Klasse von statistischen Fehlern, die modelliert werden kann erkennen,

Analyse

durch einen ergodischen stationäre Quelle mit begrenzter Speicher. "Aus diesem Grund Ansprüche Maurer, dass der Test
subsumiert eine Reihe von Standard-statistische Tests.

Der Test ist ein Kompressions-Typ-Test "auf die Idee, Ziv, dass eine universelle statistische Test basieren können auf
auf einer universellen Quelle Verschlüsselungsalgorithmus. Ein Generator sollte den Test dann, wenn die Ausgabe-Sequenz
nicht wesentlich komprimiert werden. "Laut Maurer, der
Source-Verschlüsselungsalgorithmus durch Lempel-
Ziv "zu sein scheint weniger geeignet für den Einsatz als ein statistischer Test", weil es zu schwierig sein, eine Definition scheint
Teststatistik, deren Verteilung bestimmt oder angenähert werden.

Der Test erfordert eine lange (in der Größenordnung von

10 • 2L 1000 • 2L
mit 6 = L = 16) Folge von Bits, die
gliedert sich in zwei Abschnitte der L-Bit-Blöcke (6 = L = 16), Q (
"10 • 2L) Initialisierung Blöcke und K

(
"1000 • 2L) Testblöcken. Wir nehmen

K =
nL
"
# #
$
%% & Q
um ihren Wert zu maximieren. Die Größenordnung von Q

sollten gezielt ausgewählt werden, um sicherzustellen, dass alle möglichen
L-Bit-Binär-Muster in der Tat innerhalb der auftreten
Initialisierung blockiert. Der Test ist nicht für sehr große Werte von L geeignet, da die Initialisierung braucht Zeit
exponentiellen in L.

Der Test sieht Rücken durch die gesamte Sequenz bei einem Spaziergang durch den Test Segment der L-Bit-Blöcken,
Prüfung für die nächste vorherige genaue L-Bit-Vorlage übereinstimmen und die Aufzeichnung der Strecke
log2-in Anzahl der
Blöcke zu, dass die bisherigen übereinstimmen. Der Algorithmus berechnet die

all dieser Strecken für alle L-Bit-

Vorlagen in den Test-Segment (geben, effektiv, die Anzahl der Ziffern in den binären Ausbau der einzelnen
Entfernung). Dann ist es im Durchschnitt über alle Auszugslängen durch die Anzahl der Test-Blöcken.

fn =
1K [log2i = q +1 Q + K
"
(# Indizes seit der letzten Vorkommen der i-ten Template)]

Der Algorithmus erreicht dies effizient durch Indizierung eine dynamische
Look-Up-Tabelle die Nutzung der Integer-
Darstellung der binären Bits bilden die Vorlage Blöcke. Eine standardisierte Version des Statistik -
die Standardisierung durch die vorgeschriebenen Test-im Vergleich zu einem akzeptablen Bereich basiert auf einem Standard-

Normal (Gauss) Dichte, die Nutzung der Test-Statistik ist gemein, die durch die Formel (16) in [1] gegeben ist,

EFN = 2 "L1" 1
#
$
.
Der erwartete Wert der Teststatistik fn ist, dass der Zufallsvariablen
log2G wo G = GL ist ein
geometrische Zufallsvariable mit dem Parameter
1 "2" L.
Es gibt mehrere Versionen der ungefähre empirische Formeln für die Varianz der Form

Varfn () = Cl, K () Hier

c (L, K)
stellt der Faktor, der berücksichtigt, die abhängige Art der Vorkommen von
Vorlagen. Die neueste der Näherungen (Coron und Naccache [2]: nicht in die
Test-Suite-Code eingebettet)
hat die Form

CL, K () = 0,7 "
0.8L 3-14

Analyse

Allerdings Coron und Naccache (1998) berichten, dass "die Ungenauigkeit durch [diese Näherung] können
der Test 2,67 mal mehr permissive als das, was theoretisch zugelassen. "Mit anderen Worten, das Verhältnis der
Standardabweichung von fn aus der Angleichung oben, um die wahre Standardabweichung erhalten abweicht
erheblich von einem. In Anbetracht dieser Tatsache und auch da alle Annäherungen basieren auf den
"Zulässig" Annahme, dass
Q "#
, Kann die Zufälligkeit Hypothese durch die Überprüfung der Normalität getestet werden der beobachteten Werte fn, vorausgesetzt, dass die Varianz unbekannt ist. Dies kann mit einem t-Test werden.
Die ursprüngliche Reihenfolge muss in r (partitioniert werden r =
20) Teilstrings, auf denen jeweils der Wert der
Universal-Test-Statistik ausgewertet (für den gleichen Wert von Parametern K, L und Q).
Die Stichprobenvarianz ist
ausgewertet, und die P-Wert ist

erfcfn "EL ()

varfn ()
#
$
%
%
Referenzen für Test

[1]
Ueli Maurer M., "A Universal Statistische Test für Random Bit Generators," Journal of Kryptologie. Vol. 5, Nr. 2, 1992, S. 89-105.
[2]
JS Coron und D. Naccache ", eine exakte Auswertung von Universal Test Maurer," Proceedings
von SAC '98 (Lecture Notes in Computer Science). Berlin: Springer-Verlag, 1998.
[3]
H. Gustafson, E. Dawson, L. Nielsen, W. Caelli "Ein Computer-Paket für die Messung der Stärke
von Verschlüsselungsalgorithmen, "Computers & Security. 13 (1994), S. 687-697.
[4]
AJ Menezes, PC van Oorschot, SA Vanstone, Handbook of Applied Cryptography. Boca Raton: CRC Press, 1997.
[5]
J. Ziv, "Compression, Tests auf Zufälligkeit und die Schätzung des statistischen Modells eines einzelnen
Sequenz "Sequences (ed. R. M. Capocelli). Berlin: Springer-Verlag, 1990.
[6]
J. Ziv und A. Lempel, `` A universellen Algorithmus für die sequentielle Datenkompression,"IEEE
Transactions on Information Theory. Vol. 23, S. 337-343.
3,10 lineare Komplexität Testen
Dieser Test verwendet lineare Komplexität für Zufälligkeit zu testen. Das Konzept der linearen Komplexität ist es, einen Zusammenhang
beliebter Bestandteil vieler Keystream Generatoren, nämlich Register Linear Feedback Shift (LFSR). Solch ein
Register der Länge L besteht aus L Verzögerungselemente jeweils einen Eingang und einen Ausgang. Wenn der Ausgangszustand
von LFSR ist (EL-1, ..., e1, e0), dann wird der Ausgang Sequenz (eL, eL +1, ...), erfüllt die folgenden wiederkehrenden
Formel für j =
L

"J = c1" j # 1 + c2 "c1, ..., sind cL Koeffizienten des Polynoms Verbindung zu einem gegebenen LFSR. Ein LFSR wird gesagt,
zu einem gegebenen binären Sequenz zu erzeugen, wenn dieser Sequenz ist die Ausgabe des LFSR für einige Ausgangszustand.

Analyse

Für eine gegebene Folge sn = (e1, ..., en), ist die lineare Komplexität L (sn) als die Länge des kürzesten definiert
LFSR, dass sn erzeugt wie seine ersten n Glieder. Die Möglichkeit der Verwendung der linearen Komplexität charakteristisch für
Testen Zufälligkeit beruht auf der Berlekamp-Massey-Algorithmus, der eine effiziente Möglichkeit bietet, um auf
bewerten endlichen Zeichenfolgen.

Wenn die binäre n-Sequenz sn wirklich zufällig ist, existieren Formeln [2] für die mittlere, uN = EL (sn) und die

Varianz,

"N2 = VarLsn () ()
der linearen Komplexität L (sn) = Ln, wenn die n-Sequenz sn wirklich zufällig ist.
Der Crypt-X-Paket [1] legt nahe, dass das Verhältnis in der Nähe einer Standardnormalverteilung variabel ist, so

Ln "uN () / # n
dass die entsprechenden P-Werte können von den normalen Fehler-Funktion gefunden werden. In der Tat, Gustafson et al. [1]

(S. 693) behaupten, dass "für große n, L (s86/81 Mal, dass der Standardnormalverteilung Statistik." Das ist völlig falsch. Auch
n) ist ungefähr normalverteilt mit Mittelwert n / 2 und einer Varianz verteilt
z = lsn () "
n2
#
$
%
&
"
(
der Mittelwert uN nicht asymptotisch genau so verhalten, als n / 2, und im Hinblick auf die Beschränktheit der
Varianz wird dieser Unterschied signifikant. Noch wichtiger ist, den Schwanz Wahrscheinlichkeiten der Begrenzung
Verteilung sind viel größer als die der Standard-Normalverteilung.

Die asymptotische Verteilung von
Ln "uN () / # nalong die Reihenfolge der geraden oder ungeraden Werten von n ist, dass eine

diskrete Zufallsvariable über eine Mischung von zwei geometrischen Zufallsvariablen (einer von ihnen unter erhalten
nur negative Werte). Streng genommen, gibt es nicht die asymptotische Verteilung als solche. Die Fälle n
selbst und n ungerade müssen separat mit zwei verschiedenen Grenzverteilungen aus behandelt werden.

Aufgrund dieser Tatsache die folgende Sequenz von Statistiken angepasst

Tn = "1 () NLN" #. (6)

Hier

"N =
n2 +
4 + rn18
. (7)

Diese Statistiken, die nur ganzzahlige Werte annehmen, in der Verteilung der Zufallsvariable konvergieren
PT = 0 () =
12T. This
Grenzverteilung ist nach rechts verschoben. Während für k = 1, 2, ...

PT = k () =
122k
, (8)

für k = -1, -2, ...

Es folgt aus (8), dass

P (T = k) =
122K +. (9)

PT "k> 0 () =
für k <0 (9) zeigt, dass

3-16

Analyse

PT "k () =
13 # 22kSo die
"= Tobs [] +1 P-Wert entsprechend dem beobachteten Wert Tobs kann in der folgenden Weise ausgewertet werden. Lassen Sie
. Dann ist die P-Wert ist
.

13 "22 # $ 1 +
13 "22in Blick auf die diskrete Natur von dieser Verteilung und die Unmöglichkeit der Verwirklichung des einheitlichen
Verteilung für P-Werte, kann die gleiche Strategie verwendet, der mit anderen Tests in dieser Situation verwendet werden.
Nämlich Partition der String der Länge n, so dass n = MN, in N Teilstrings jeweils der Länge M. Für die
Test auf der Grundlage der linearen Komplexität Statistik (6), zu bewerten TM innerhalb der j-ten substring der Größe M, und
Wählen Sie K +1 Klassen (je nach M.) Für jede dieser Zeichenketten, die Frequenzen, .0, .1, ...,. K, der
Werte von TM Zugehörigkeit zu einem der K +1 gewählt Klassen, .0 + .1 + ... +. K = N, bestimmt. Es ist
zweckmäßig, die Klassen mit den Endpunkten bei semi-Zahlen zu wählen.

Die theoretischen Wahrscheinlichkeiten p0, p1, ..., pK dieser Klassen sind aus bestimmt (8) und (9). Aus diesem
Dazu hat M groß genug sein für die Grenzverteilung von (8) gegeben und (9) eine Verfügung
vernünftige Annäherung. M soll mehr als 500. Es wird empfohlen, M gewählt werden, dass 500 =
M =
5000.

Die Frequenzen werden durch den verwachsenen
"2-Statistik

"2 =
I $ N%% ii = 0K
&
.
, die unter die Zufälligkeit Hypothese, hat eine ungefähre
"2-Verteilung mit K Freiheitsgraden.
Die ausgewiesenen P-Wert ist

e "u/2uK/2 # 2 (obs)
$%
& K / 2 () 2K / 2 =.
Nach wie vor eine konservative Bedingung für die Nutzung der
"2-Näherung ist, dass
Nmini

"I # 5.
Für einigermaßen große Werte von M und N, die folgenden Klassen (K = 6) scheinen angemessen zu sein: {T =
-2,5},
{-2,5 <T =
-1,5}, {-1,5 <T =
-0,5}, {-0,5 <T =
0,5}, {0,5 <T =
1,5}, {-1,5 <T =
2.5}, {T> 2,5}.

Die Wahrscheinlichkeiten dieser Klassen sind p0 = 0,01047, p1 = 0,03125, p2 = 0,12500, p3 = 0,50000, p4 = 0,25000,
p5 = 0,06250, p6 = 0,020833. Diese Wahrscheinlichkeiten sind wesentlich verschieden von denen, die aus den erhaltenen
normale Annäherung, für die ihre Zahlenwerte sind: 0,0041, 0,0432, 0,1944, 0,3646, 0,2863,
0,0939, 0,0135.

3-17

Analyse

Referenzen für Test

[1]
H. Gustafson, E. Dawson, L. Nielsen, und W. Caelli (1994), "Ein Computer-Paket für die Messung
die Stärke des Verschlüsselungsalgorithmus, "Computers and Security. 13, S. 687-697.
[2]
AJ Menezes, PC van Oorschot und SA Vanstone (1997), Handbook of Applied Cryptography. CRC Press, Boca Raton, FL.
[3]
R.A. Rueppel, Analyse und Konzeption von Stream Ciphers. New York: Springer, 1986.
3,11 Seriell-Test
Die (verallgemeinerte) serial-Test stellt eine Reihe von Verfahren zur Prüfung der Gleichmäßigkeit der Basis
Verteilungen von Mustern bestimmter Länge.

Insbesondere für
"I1Kimi1, ..., im Durchlaufen der Menge aller möglichen 2m 0,1 Vektoren der Länge m, lassen
bezeichnen die Häufigkeit des Musters (i1, ..., im) in der "zirkularisiert" Folge von Bits (e1, ..., en, e1, ...,

em-1).

Set

"M2 =
2mn
I1Kim
%
&
"
+ So

"M2
ist ein
"2-Typ Statistik, aber es ist ein verbreiteter Irrtum anzunehmen, dass die
"M2
hat die
"2-Verteilung.

In der Tat, die Frequenzen

"I1Kim
sind nicht unabhängig.

Die entsprechende generalisierte serielle Statistiken für die Prüfung von Zufälligkeit ([1], [2] und [3]) sind

und

"# M2 = # m2 $ # $ m" 2 # m2 = # m2 2 $ # (hier
"02 =" # 12 = 0). Dann

"# M2
hat eine
"2-Verteilung mit 2m-1 Freiheitsgraden und
"2 # m2
hat eine

"2
-Verteilung mit 2m-2 Freiheitsgraden. So für kleine Werte von m,
m "log2n ()#$% 2, kann man

finden die entsprechenden 2m P-Werte aus der Standard-Formeln.

P "Wert1 = igamc2mP" Wert2 = igamc2mThe Ergebnis für

"Nr. 22
und die übliche Zählung der Frequenzen ist falsch in [1, S. gegeben 181, Formel

(5,2)]: +1 sollte mit -1 ersetzt werden.

Die Konvergenz von

"# M2
die
"2-Verteilung wurde von Good (1953) bewiesen.

3-18

Analyse

Referenzen für Test

[1] IJ Good (1953), "Der serielle Test für die Anzahl der Proben und anderen Tests auf Zufälligkeit", Proc.
Cambridge Philos. Soc .. 47, S. 276-284.
[2] M. Kimberley (1987), "Vergleich von zwei statistischen Tests für Keystream Sequenzen," Electronics
Letters. 23, S. 365-366.
[3] DE Knuth (1998), The Art of Computer Programming. Vol. 2, 3. Aufl.. Reading: Addison-
Wesley, Inc., S. 61-80.
[4] AJ Menezes, PC van Oorschot und SA Vanstone (1997), Handbook of Applied Cryptography. Boca Raton, FL: CRC Press, S. 181.
3,12 Ungefähre Entropy-Test

Ungefähre Entropie Eigenschaften [1] auf sich wiederholende Muster in der Zeichenfolge. WENN
Yim () = "i, K," i +, Set
Cim =
1n +1 "m # und

"(M) =
1n +1 # mlogCimi +,
Cim
ist die relative Häufigkeit des Auftretens des Musters Yi (m) in den String und-F (m) ist die Entropie

die empirische Verteilung, die sich auf die Menge aller 2m mögliche Muster der Länge m,

"(M) = # # LLOG 12m
$
,

wo pl ist die relative Häufigkeit der Muster l = (i1, ..., im) in der Zeichenfolge.
Die ungefähre Entropie ApEn der Ordnung m, m =
1 ist definiert als

ApEnm ()="(m) mit ApEn (0) =-F (1). "ApEn (m) Maßnahmen der logarithmischen Frequenz, mit der Blöcke der Länge m, die
nahe beieinander bleiben nahe beieinander für die Blöcke um eine Position erweitert. So kleine Werte von
ApEn (m) bedeuten starke Regelmäßigkeit, oder Persistenz, in einer Sequenz. Alternativ große Werte von ApEn (m)
implizieren erhebliche Fluktuation, oder Unregelmäßigkeiten. "[1, S. 2083].

Pincus und Kalman [2] definierten Reihenfolge zu m-unregelmäßig (m-random), wenn seine ungefähre Entropie
ApEn (m) nimmt den größtmöglichen Wert. Sie bewerteten Mengen
2
3ApEn (m), m = 0, 1, 2 für Binär-und
Zur 3. Erweiterungen der e, p, und

mit dem überraschenden Ergebnis, dass der Ausbau der
zeigte mehr Unregelmäßigkeiten als die von S.
3-19

1.

Analyse

Für eine feste Blocklänge m, sollte man erwarten, dass in langen random

(unregelmäßig) Streicher,

ApEn (m) "log2.

Die Begrenzung der Verteilung von

nlog2 "ApEn (deckt sich mit einer

"2-Zufallsvariable mit 2m

Freiheitsgrade. Diese Tatsache bildet die Grundlage für einen statistischen Test, wie

Rukhin gezeigt wurde [3].

So mit

"2 (obs) = nlog2 ist die berichtete P-Wert

igamc2m "1, # 2 (obs) /.
Eigentlich ist diese Begrenzung Verteilung der ungefähre Entropie genauer für seine modifizierte Definition als

"
~ (M)
= # I1Kimlog $
,
WO
"I1Kimdenotes die relative Häufigkeit der Vorlage (i1, ..., im) in die erweiterte (oder kreisförmig)
Version des Original-String, dh in den String
"1, K," n ", 1, K," m. Lassen

"I1Kim = n # i1Kim
werden die

Frequenz des Musters i1, ..., im. Unter unserer Definition
"I1Kim =" i1Kimkk #
, So dass für jeden
.

m,
i1Kim
"= N
Definieren Sie die geänderte ungefähre Entropie als

Apen (m)
~

= "
~ (M)
.
Mit Jensen-Ungleichung,
logs "Apen (m)
~
für alle m, während es möglich ist, dass
logs <Apen (m)
.

Daher ist der größtmögliche Wert der modifizierten Entropie nur log s, die, wenn n = erreicht ist
sm, und die Verteilung aller m-Muster einheitlich ist. Bei der Berechnung der ungefähren Entropie für mehrere
Werte von m, ist es sehr bequem, die Summe aller Frequenzen von m-Vorlagen haben gleich n.

Wenn n groß ist, kann ApEn (m) und seine modifizierte Version unterscheidet sich nur wenig. Tatsächlich hat man mit
"# I1Kim = (n $ m +" # i1Kimi1Kim
$ = N% m + und
"I1Kim
$ "I1Kim
% Daraus folgt, dass
"I1Kim
$ "I1Kim
%
m "(m)
und
"
~ (M)
was darauf schließen lässt, dass für eine feste

muss in der Nähe für große n. Daher Pincus 'ungefähre
Entropie und seine modifizierte Version sind auch in der Nähe, und ihre asymptotische Verteilungen müssen übereinstimmen.

Referenzen für Test

[1]
S. Pincus und BH Singer, "Zufall und Grad der Unregelmäßigkeit", Proc. Natl. Acad. Sci. USA. Vol. 93, März 1996, S. 2083-2088.
3-20

Analyse

[2]
S. Pincus und RE Kalman, "Nicht alle (möglicherweise)" random Proc "-Sequenzen gleich sind, geschaffen."
Natl. Acad. Sci. USA. Vol. 94, April 1997, S. 3513 bis 3518.
[3]
A. Rukhin (2000), "Ungefähre Entropie für die Prüfung Zufälligkeit", Journal of Applied Probability. Vol. 37, 2000.
3,13 Summenwerte (Cusum) Test
Dieser Test basiert auf der maximalen absoluten Wert der Partialsummen der Sequenz dargestellt in der
± 1 Mode. Große Werte dieser Statistik zeigen, dass es entweder zu viele Einsen oder zu viele Nullen am
den frühen Phasen der Sequenz. Kleine Werte zeigen an, dass Einsen und Nullen zu werden vermischt
"S
k = Xn + K + Xnevenly. A

Dual-Test kann von der umgekehrten Zeit Irrfahrt mit abgeleitet werden. Mit diesem

Definition, die Interpretation der Testergebnisse ist durch Austausch "der Anfangsphase" von dem verstorbenen geändert
Stufen. "

Der Test basiert auf der Begrenzung der Verteilung des Maximums der absoluten Werte der Partialsummen basiert,
max1 "k" NSK,
limn "#
Pmax1 $ k $ nSkn
%
&
"
=
4
"
(# 1) J2J ein expj = 0
* +
(10)
Mit der Teststatistik
z = max1 "k" NSK (die Zufälligkeit Hypothese ist für große Werte abgelehnt
von z, und die entsprechende P-Wert ist
1 "Hmax1 # k # NSK (wo die Funktion G (z) ist durch die Formel (11) definiert.

Die Serie H (z) in der letzten Zeile (10) konvergiert schnell und sollte für die numerische Berechnung verwendet werden
nur für kleine Werte von z Die Funktion G (z) (die gleich ist
max1 "k" NSK (obs) / z) für alle z) ist bevorzugt, dass die
Berechnung für moderate und große Werte von,

G (z) =

12 "
Zz,
= ("1) k # 2k + k =" $
$
%
= "(Z)#"(# z)+="(z)#"(# z)#"#(z)$#($ z) $" 1 #
42 $ zexp%
&
"
(11)
wo F (x) ist die Standard-Normalverteilung.

3-21

Analyse

Mehr direkt über Theorem 2.6, S. 17 von Revesz [2], so erhält man

Pmax1 "k" NSK # z ()
= 1 "P4K" k = "#
#
$
+ P4K +1 () z <k = "#
#
$
.

Diese Formel gilt für die Bewertung der P-Werte verwendet mit

z = max1 "k" NSK (obs) / Die Zufälligkeit Hypothese ist für große Werte von z abgelehnt

Referenzen für Test

[1]
Frank Spitzer, Principles of Random Walk. Princeton: Van Nostrand, 1964 (besonders S. 269).
[2]
Pal Revesz, Random Walk in Random und Non-Random-Umgebungen. Singapore: World
Scientific, 1990.

3,14 Zufällige Ausflüge Testen

Dieser Test basiert auf Berücksichtigung aufeinanderfolgenden Summen der binären Bits (plus oder minus einfache) als eindimensionale Basis
random walk. Der Test erkennt Abweichungen von der Verteilung der Anzahl der Besuche der
Irrfahrt bis zu einem gewissen "Zustand", dh einen beliebigen ganzzahligen Wert.

Betrachten Sie die Random-Walk-Sk = X1 + ... + Xk als eine Folge von Exkursionen zu und von Null

i, k, l ():. Si "1 = Sl + J sei bezeichnen die Gesamtzahl der solche Ausflüge in die Zeichenfolge die Grenzverteilung für diese (random)
Zahl J (dh die Anzahl der Nullen unter den Summen Sk, k = 1, 2, ..., n, wenn S0 = 0) ist bekannt, dass

limn "#
PJN <z
$
%
&

"

(

) =

.

(12)

Der Test weist die Zufälligkeit Hypothese sofort, wenn J ist zu klein, dh, wenn die folgenden P-Wert ist
klein:

PJ <J (obs) () "

.

WENN
J <max (0.005n, ist die Zufälligkeit Hypothese abgelehnt. Ansonsten wird die Anzahl der Besuche der
random walk S zu einem bestimmten Zustand ausgewertet wird.

3-22

Analyse

Lassen. (X) die Anzahl der Besuche auf x, x sein.
0, während einer 0-Ausflug. Die Verteilung ist in Revesz abgeleitet

[2] und Baron und Rukhin [1]:
und für k = 1, 2, ...

P "(x) = 0 () = 1 #
P "(x) = k () =

14x21 (13)
(14)
Dies bedeutet, dass (x) = 0 mit Wahrscheinlichkeit 1 -1 / 2 | x |;. Sonst (mit Wahrscheinlichkeit 1 / 2 | x |), (x) fällt mit.
eine geometrische Zufallsvariable mit dem Parameter 1 / 2 | x |.

Es ist leicht zu sehen, dass

E "(x) = 1,
und

Var "(x) () = 4xA nützliche Formel lautet:

P "(x) # a +1 () = 15)
Die obigen Ergebnisse sind auf Zufälligkeit Prüfung in der folgenden Weise verwendet. Für eine "repräsentative"-Kollektion
der x-Werte (sagen wir, 1 =
x =
7 oder -7 =
x =
-1: -4 =
x =
4 in die Test-Suite-Code verwendet wird), bewerten die beobachteten Frequenzen. k (x) die Zahl k der Besuche in den Zustand x in J Ausflüge, die im String vorkommen. So

"K (x) =" kj (j = 1 J #
mit
"Kj (x) = 1exactly gleich k, und
"Kj (x) = 0otherwise. Pool der Werte. (X) in Klassen, sagen wir, k = 0, 1, ..., 4 mit
wenn die Zahl der Besuche auf x während der j-ten Ausflug (j = 1, ..., J) ist

eine zusätzliche Klasse k =
5. Die theoretischen Wahrscheinlichkeiten für diese Klassen sind:

"0 = P (# (x) = 0) =" k = P (# (x) = k) =
"5 = P (# (x) $ 5) =
Diese Wahrscheinlichkeiten haben die Form

Analyse

p0 (x) p1 (x) p2 (x) p3 (x) p4 (x) p5 (x)
x = 1 0,5000 0,2500 0,1250 0,0625 0,0312 0,0312
x = 2 0,7500 0,0625 0,0469 0,0352 0,0264 0,0791
x = 3 0,8333 0,0278 0,0231 0,0193 0,0161 0,0804
x = 4 0,8750 0,0156 0,0137 0,0120 0,0105 0,0733
x = 5 0,9000 0,0100 0,0090 0,0081 0,0073 0,0656
x = 6 0,9167 0,0069 0,0064 0,0058 0,0053 0,0588
x = 7 0,9286 0,0051 0,0047 0,0044 0,0041 0,0531

Vergleichen Sie diese Frequenzen um die theoretischen mit der .2-Test,

"2 (x) =
K (k = 05
&
,
, die für jedes x unter der Zufälligkeit Hypothese, muss etwa ein .2-Verteilung mit 5

Freiheitsgrade. Dies ist eine gültige testen, wenn

Jmin "k (x) # 5, d. h., wenn J =
500. (Die Test-Suite-Code verwendet

p4 (x = 4) für min pk (x).) Wenn diese Bedingung nicht gilt, Werte. (x) muss in größeren Klassen zusammengelegt werden.
Die entsprechende Batterie von P-Werte berichtet. Diese Werte sind aus der Formel

1 "P52,
2 (obs) (2
$
%
&
.
Referenzen für Test

[1] M. Baron und AL Rukhin, "Verteilung der Anzahl der Visits Für einen Random Walk" Communications in Statistics: Stochastic Models. Vol. 15, 1999, S. 593-597.
[2] Pal Revesz, Random Walk in Random und Non-Random-Umgebungen. Singapore: World Scientific,
1990.
[3] Frank Spitzer, Principles of Random Walk. Princeton: Van Nostrand, 1964, (besonders S. 269).
3,15 Zufällige Ausflüge Variant-Test
Eine Alternative zu der zufälligen Ausflüge Test kann wie folgt abgeleitet werden. Mit der Notation des vorherigen
Unterabschnitt lassen. J (x) die Gesamtzahl der Besuche in x bei J Exkursionen werden. (Die Test-Suite wird davon ausgegangen, J
=
500.) Da Sk erneuert bei jedem Null. J (x) ist eine Summe von unabhängigen, identisch verteilten Variablen mit
die gleiche Verteilung wie. (x) = .1 (x). Daher ist die Grenzverteilung. J (x),

limJ "#
P
$ J (x)%% &
"
(
(
z),
ist normal. Die Zufälligkeit Hypothese wird abgelehnt, wenn der P-Wert

Analyse

erfc
"J (x) (obs) # $
%
&
&
ist klein.

Referenzen für Test

[1]
M. Baron und AL Rukhin, "Verteilung der Anzahl der Visits Für einen Random Walk"
Communications in Statistics: Stochastic Models. Vol.15, 1999.
[2]
Pal Revesz, Random Walk in Random und Non-Random-Umgebungen. Singapore:
World
Scientific, 1990.
[3]

Frank Spitzer, Principles of Random Walk. Princeton: Van Nostrand, 1964 (besonders S. 269).
3-25

Analyse

4. Testing Strategie und Ergebnis Interpretation
Drei Themenbereiche werden in diesem Abschnitt behandelt werden: (1) Strategien für die statistische Analyse einer zufälligen
Number Generator, (2) die Interpretation der empirischen Ergebnisse mit dem NIST Statistische Test Suite, und (3)
allgemeine Empfehlungen und Richtlinien.

4.1 Strategien für die statistische Analyse der RNG
In der Praxis gibt es viele unterschiedliche Strategien in der statistischen Analyse einer zufälligen Zahl der Beschäftigten
Generator. NIST hat sich die Strategie in Abbildung 1 dargestellt angenommen. Abbildung 1 gibt einen architektonischen
Darstellung der fünf Etappen in der statistischen Prüfung von einem Zufallsgenerator beteiligt.

Etappe 1: Auswahl eines Generators

Wählen Sie ein Hardware-oder Software-basierter Generator zur Auswertung. Der Generator sollte eine binäre
Folge von 0 und 1 ist von einer bestimmten Länge n. Beispiele für Pseudo-Generatoren (PRNG), dass sein
gewählt sind ein DES-basierte PRNG von ANSI X9.17 (Anhang C), sowie zwei weitere Methoden, die
angegeben in FIPS 186 (Anlage 3) und basieren auf den Secure Hash Algorithm

(SHA-1) und die Daten auf
Encryption Standard (DES).

Stufe 2: Binary Sequence-Generation

Für eine feste Folge der Länge n und die vorgewählte Generator, bauen eine Reihe von m binären Sequenzen und
Speichern Sie die Sequenzen zu einem file7.

Stufe 3: Führen Sie das Statistische Test Suite

Rufen Sie die NIST Statistische Test Suite unter Verwendung der Datei in Stufe 2 und die gewünschte Sequenz Länge hergestellt.
Wählen Sie das statistische Tests und entsprechenden Input-Parameter (zB Blocklänge) angewendet werden.

Stufe 4: Untersuchen Sie die P-Werte

Eine Ausgabedatei wird durch die Test-Suite mit den entsprechenden Zwischen-Werte, wie zB Test-Statistiken generiert werden,
und P-Werte für die einzelnen statistischen Tests. Basierend auf diesen P-Werte, eine Aussage über die Qualität der
Sequenzen hergestellt werden kann.

Stage 5: Beurteilung: Pass / Fail Assignment

Für jeden statistischen Test, ist ein Satz von P-Werten (entsprechend der Menge von Sequenzen) produziert. Für eine feste
Signifikanzniveau, einen bestimmten Prozentsatz der P-Werte sollen Fehler hinweisen. Zum Beispiel, wenn die
Signifikanzniveau gewählt zu 0,01 sein (also ein
= 0,01), dann etwa 1% der Sequenzen wird erwartet, dass

scheitern. Eine Sequenz übergibt einen statistischen Test, wenn der P-Wert = a und nicht anders. Für jede
statistischen Test, der Anteil der Sequenzen, die passieren berechnet wird und entsprechend analysiert. Mehr vertiefte
Analyse sollte mit zusätzlichen statistischen Verfahren (siehe Abschnitt 4.2.2) werden.

7 Beispiel Daten können auch von Random Number George Marsaglia der CDROM, erhältlich bei
http://stat.fsu.edu/pub/diehard/cdrom/.

4.2 Die Interpretation der empirischen Ergebnisse
Drei Szenarien sind typisch Ereignisse, die durch empirische Tests auftreten können. Fall 1: Die Analyse der Pvalues
kein Hinweis auf eine Abweichung von der Zufälligkeit. Fall 2: Die Analyse zeigt deutlich eine Abweichung
von Zufälligkeit. Fall 3: Die Analyse ist nicht schlüssig.

Die Interpretation der empirischen Ergebnisse können in einer beliebigen Anzahl von Arten durchgeführt werden. Zwei Ansätze NIST
hat angenommen umfassen (1) die Prüfung der Anteil der Sequenzen, die eine statistische Test bestehen und (2)
die Verteilung der P-Werte, die für Einheitlichkeit zu überprüfen.

Für den Fall, dass einer dieser beiden Ansätze nicht (dh, die entsprechende Nullhypothese muss abgelehnt werden),
zusätzliche numerische Experimente sollten an verschiedenen Proben des Generators

durchgeführt werden, um festzustellen,
ob das Phänomen wurde eine statistische Anomalie oder ein klarer Beweis der Nicht-Zufälligkeit.

4.2.1 Anteil der Sequenzen eine Prüfung besteht,
Abbildung 4-1: P-Wert Plot
Angesichts der empirischen Ergebnisse für einen bestimmten statistischen Test, Berechnung des Anteils der Sequenzen, die passieren. Für
Beispielsweise wenn 1000 Binärsequenzen getestet wurden (dh m =
1000), ein
= 0,01 (Signifikanzniveau), und 996 binäre
Sequenzen hatte P-Werte =
.01, Dann ist das Verhältnis
996/1000 = 0,9960.

Die Palette der akzeptablen Verhältnis wird mit
3p
± das Konfidenzintervall definiert als,,

WO

$ P
= 1-a, und m ist die Stichprobengröße. Wenn der Anteil
außerhalb dieses Intervalls, dann gibt es Hinweise, dass die
Daten werden nicht zufällig. Beachten Sie, dass andere Werte der Standardabweichung verwendet werden könnten. Für das obige Beispiel, die

(Dh, der Anteil sollte über 0,9805607 liegen. Diese

Konfidenzintervall

kann dargestellt mit Hilfe eines Graphen wie in Abbildung 4-1 gezeigt werden. Das

Konfidenzintervall wurde mit einem
Normalverteilung als Approximation der Binomialverteilung, die einigermaßen genau ist für
große Stichproben (z. B., n =
1000).

4.2.2 gleichmäßige Verteilung der P-Werte
Abbildung 4-2: Histogramm der P-Werte
Die Verteilung der P-Werte untersucht, um sicherzustellen,
Uniformität. Dies kann visuell dargestellt werden mit einem
Histogramm (siehe Abbildung 4-2), wobei das Intervall
zwischen 0 und 1 ist in 10 Teilintervalle unterteilt, und die
P-Werte, die in jedem Teilintervall liegen, sind gezählt
und angezeigt.

Uniformity kann auch über einen Antrag der bestimmt werden
ein 0,2-Test und die Bestimmung eines P-Wert
entsprechend der Goodness-of-Fit-Test Verteilungswirkungen
auf die P-Werte für eine beliebige statistische Test erhalten
(D. h., ein P-Wert des P-Werte). Dies wird erreicht,

Analyse

durch die Berechnung =
#, Wobei Fi ist die Zahl der P-Werte in Teilintervall i und s ist die
Stichprobengröße. Ein P-Wert ist so, dass berechnete P-valueT = igamc (
!
. Wenn P-valueT =

0,0001, dann

Die Sequenzen können als gleichmäßig verteilt werden.

4.3 Allgemeine Empfehlungen und Richtlinien
In der Praxis kann es viele Gründe gegeben, warum ein Datensatz einen statistischen Test nicht bestanden hat. Die folgenden
ist eine Liste der möglichen Erklärungen. Die Liste wurde auf der Grundlage NIST statistischen Tests Bemühungen zusammengestellt.

(A) Ein falsch programmiert statistischen Test.
Sofern nicht anders angegeben, ist davon auszugehen, dass ein statistischer Test zugeschnitten war an einem Griff werden
Besonders problematisch Klasse. Da die NIST-Test-Code wurde geschrieben, um die Auswahl der damit
Eingabeparameter, wurde der Code in eine beliebige Anzahl von Möglichkeiten, verallgemeinert werden. Leider ist this
nicht zwangsläufig zur Codierung einfach zu übersetzen.

Ein paar statistische Tests wurden mit künstlichen obere Schranken gezwungen worden. Zum Beispiel, die
random Ausflüge Tests sind davon ausgegangen, dass nicht mehr als max {1000, n/128} Zyklen.
Denkbar ist, kann feste Parameter erhöht werden, je nach experimentellen Bedingungen.

(B) Ein unterentwickeltes (unreifen) statistischen Test.
Es gibt Gelegenheiten, bei denen entweder die Wahrscheinlichkeit oder die Komplexität der Theorie nicht ausreichend entwickelt
oder versteht man eine strenge Analyse des statistischen Tests zu erleichtern.

Im Laufe der Zeit sind statistische Tests im Lichte der neuen Ergebnisse überarbeitet.

Da viele statistische Tests
basieren auf asymptotischen Näherungen basiert, erfordert eine sorgfältige Arbeit zu leisten, wie zu bestimmen
gute Näherung ist.

(C) Eine fehlerhafte Umsetzung von einem Zufallsgenerator.
Es könnte sein, plausibel, dass ein Hardware-RNG oder ein Software-RNG hat aufgrund eines Fehlers in der gescheiterten
Design oder aufgrund einer Codierung Umsetzung Fehler. In jedem Fall muss eine sorgfältige Überprüfung vorgenommen werden
dies auszuschließen.

(D) Falsch geschriebene Codes nutzen Test Eingabedaten.
Ein weiterer Bereich, unter die Lupe genommen werden muss, ist die Nutzung von Testdaten. Die Testdaten
produziert von einem (P) RNG müssen, bevor sie durch einen statistischen Test verwendete verarbeitet werden. Für
beispielsweise die Verarbeitung könnte zählen die Aufteilung des Ausgabe-Stream aus dem (P) RNG in
entsprechende großen Blöcken, sowie der Umsetzung der 0 ist, um negative. Gelegentlich war es
festgestellt, dass der Fehler von einem statistischen Test aufgrund von Fehlern im Code verwendet werden, um wurden
Verarbeitung der Daten.

Analyse

(E) Schlechte mathematische Routinen zur Berechnung von P-Werten

Qualität Mathematik-Software muss verwendet werden, um ausgezeichnete Annäherungen zu gewährleisten, wann immer möglich sein.
Insbesondere ist die unvollständige Gamma-Funktion erschwert die Angleichung für größere
Werte der Konstanten a. Schließlich wird P-Wert Formeln in falsche Werte durch Ergebnis
Schwierigkeiten, die sich aus der numerischen Approximation. Um die Wahrscheinlichkeit dieses Ereignisses zu verringern,
NIST hat es vorgezogen Eingabeparameter vorgeschrieben.

(F) Falsche Entscheidungen für Eingabeparameter.
In der Praxis wird ein statistischer Test liefert keine verlässlichen Ergebnisse für alle scheinbar gültige Eingabe
Parameter. Es ist wichtig zu erkennen, dass Einschränkungen auf Tests basieren auf einem Test-by-Test gemacht
Basis. Nehmen Sie die ungefähre Entropy-Test, zum Beispiel. Für eine Sequenz Länge in der Größenordnung
von 106, würde man erwarten, dass Blocklängen nähert log2 n akzeptabel wäre.
Leider ist dies nicht der Fall. Empirische Untersuchungen zeigen, dass über m = 14, die beobachtet Teststatistik beginnen, mit dem erwarteten Wert nicht einverstanden (insbesondere bei bekannten
gute Generatoren, wie SHA-1). Daher können bestimmte statistische Tests empfindlich auf Input
Parameter.

Überlegungen müssen oft in Bezug auf die numerische Experimente eingegeben werden
Parameter, nämlich: Sequenzlänge, Stichprobengröße, Blockgröße und Vorlage.

Sequence Length

Die Bestimmung, wie lange Sequenzen sollten für die Zwecke der statistischen

genommen werden
Testen ist schwierig, Adresse. Wenn man die FIPS 140-1 statistische Tests untersucht, ist es offensichtlich,
dass Sequenzen sollten etwa 20.000 Bit lang.

Allerdings ist die Schwierigkeit, mit der Einnahme relativ kurze Sequenz Längen problematisch in der
Sinne, dass einige statistische Tests, wie Universal Statistische Maurer Test, erfordern extrem
lange Sequenz Längen. Einer der Gründe ist die Erkenntnis, dass asymptotische Approximationen
werden bei der Bestimmung der Grenzverteilung. Aussagen über die Verteilung für bestimmte Teststatistiken sind schwieriger zu Adresse für kurze Länge Sequenzen als ihre mehr
Länge Pendants.

Stichprobenumfang

Die Frage der Stichprobengröße ist die Wahl des Signifikanzniveaus gebunden. NIST empfiehlt
dass für diese Tests, sollte der Benutzer fix das Signifikanzniveau auf mindestens 0,001 sein, aber nicht größer
als 0,018. Eine Stichprobe, die in keinem Verhältnis zu dem Signifikanzniveau ist möglicherweise nicht geeignet.
Zum Beispiel, wenn das Signifikanzniveau (a) gewählt bis 0,001, dann ist zu erwarten, dass 1
von 1000 Sequenzen werden abgelehnt. Wenn eine Stichprobe von nur 100 Sequenzen ausgewählt ist, es
wäre selten zu einer Ablehnung zu beobachten. In diesem Fall kann die Schlussfolgerung gezogen werden, dass ein
Generator produzierte zufälligen Sequenzen, wenn aller Wahrscheinlichkeit nach eine ausreichend große

genug Probe wurde nicht verwendet. So sollte die Probe auf die Reihenfolge der Kehrwert der sein
Signifikanzniveau (a
-1). Das heißt, für ein Niveau von 0,001, sollte eine Probe mindestens 1000 Sequenzen. Im Idealfall sollten viele verschiedene Proben analysiert werden.

8 Hinweis, dass für FIPS 140-2, das Signifikanzniveau auf 0,0001 wurde für die Macht up-Tests eingestellt.

Analyse

Blockgröße

Blockgrößen sind abhängig von den einzelnen statistischen Tests. Im Fall der Maurer Universal
Statistische Tests, Reichweite Blockgrößen von 1 bis 16. Doch für jede spezifische Blockgröße, ein
minimale Sequenzlänge verwendet werden sollte. Wenn die Blockgröße wurden bei 16 festgelegt, eine Folge von
mehr als eine Milliarde Bits erforderlich wären. Für einige Benutzer kann das nicht machbar sein.

Intuitiv scheint es, dass je größer die Blockgröße, desto mehr Informationen gewonnen werden konnten
aus der Analyse einer Sequenz, wie in der ungefähren Entropy-Test. Doch ein Block Größe, die zu groß ist, sollte auch nicht gewählt werden, denn sonst werden die empirischen Ergebnisse können
irreführend und falsch, weil die Teststatistik ist besser, von einer ausgeprägten

angenähert
Wahrscheinlichkeitsverteilung.
! Praxis, berät NIST Auswahl einer Blockgröße nicht größer als

, Wobei n die Sequenzlänge. Allerdings halten einige Ausnahmen, und damit NIST

schlägt die Wahl eines kleineren Blockgröße.

Schablone

Bestimmte statistische Tests zur Erkennung von globalen Nicht-Zufälligkeit geeignet. Aber auch andere
statistischen Tests sind eher geneigt auf die Beurteilung lokaler Nicht-Zufälligkeit, wie Tests entwickelt, um
das Vorhandensein von zu vielen m-Bit-Muster in einer Sequenz. ! Macht es Sinn, dass

Vorlagen von einer Blockgröße von mehr als

sollte nicht gewählt werden, da Häufigkeitszählungen

wird sehr wahrscheinlich in der Nähe von Null, ist das keine brauchbaren Informationen. Daher müssen entsprechende Entscheidungen getroffen werden.

Andere Überlegungen

Im Prinzip gibt es viele häufig auftretende Fragen bezüglich Zufälligkeit Tests. Vielleicht ist die am häufigsten gestellte Frage lautet: "Wie viele Tests sollte man sich bewerben?" In
Praxis kann niemand wirklich beantworten diese Frage. Der Glaube ist, dass die Tests sollten
unabhängig von einander so weit wie möglich.

Eine weitere, häufig gestellte Frage betrifft die Notwendigkeit für die Anwendung eines monobits Test (dh
Frequency-Test), anstelle von Universal Statistische Maurer-Test. Die Wahrnehmung ist, dass Maurer
Universal-Statistischer Test ersetzt die Notwendigkeit, einen monobits Test anwenden. Dies kann zutreffen für
unendlich lange Sequenzen. Allerdings ist es wichtig, im Hinterkopf behalten, dass es sein
Fälle, in denen eine endliche binäre Folge wird Maurer Universal Statistische Test zu bestehen, aber nicht
die monobits testen. Aufgrund dieser Tatsache empfiehlt NIST, dass die Frequenz-Test angewendet werden
zuerst. Wenn die Ergebnisse dieses Tests die Nullhypothese zu unterstützen, dann kann der Benutzer gehen zu gelten
andere statistische Tests.

4.4 Anwendung des Multiple Tests

Bei einer Besorgnis über die Anwendung von mehreren Tests durchgeführt NIST eine Studie zur Bestimmung der
Abhängigkeit zwischen den Tests. Die Durchführung der Tests wurde durch Verwendung eines überprüft Kolmogorov-
Smirnov-Test der Einheitlichkeit auf die P-Werte aus den Sequenzen erhalten. Allerdings benötigt es eine
Annahme, dass die Sequenzen, um zu testen, Uniformität erzeugt wurden, ausreichend zufällig waren. Es GIBT
viele Tests in der Suite. Einige Tests sollten intuitiv zu geben unabhängige Antworten (zB die Frequenz-Test
und Testläufe, dass die Bedingungen auf den Frequenzen sollte komplett verschiedene Aspekte des Zufalls zu beurteilen).
Andere Tests, wie die Cusum Test und der Test läuft, führen in P-Werte, die wahrscheinlich korreliert werden.

Analyse

Um zu verstehen, die Abhängigkeiten zwischen den Tests, um redundante Tests zu beseitigen und um sicherzustellen, dass
die Tests in der Suite sind in der Lage, eine angemessene Auswahl von gemusterten Verhaltensweisen, eine Faktorenanalyse der Erkennung
resultierenden P-Werte wurde durchgeführt. Genauer gesagt, um die Unabhängigkeit zu beurteilen, m-Sequenzen
binäre Pseudo-Ziffern generiert wurden, die jeweils der Länge n und alle k = 161 Tests in der Suite wurden angewandt
diejenigen Sequenzen, ihre Zufälligkeit zu bestimmen. Jeder Test produzierte eine Bedeutung Wahrscheinlichkeit, bezeichnen
von pij die Bedeutung Wahrscheinlichkeit Test i auf Sequenz j.

Angesichts der gleichmäßig verteilten pij, die Transformation

führt zu normalverteilten

Variablen. Lassen zj den Vektor der transformierten Bedeutung Wahrscheinlichkeiten entsprechend der = it Sequenz sein.
Eine Hauptkomponentenanalyse wurde am z1 durchgeführt, ..., zm. In der Regel eine kleine Zahl von
Komponenten genügt, um einen großen Anteil an der Variabilität erklären, und die Anzahl dieser Komponenten
kann verwendet werden, um die Anzahl der "Dimensionen" der nonrandomness durch die Suite Tests gespannt zu quantifizieren. Sterben
Principal Component Analyse dieser Daten durchgeführt wurde. Diese Analyse Extrakte 161 Faktoren, gleich der

Anzahl der Tests. Der erste Faktor ist der, dass die größte Variabilität erklärt. Wenn viele Tests sind
korreliert, wird ihre P-Werte stark von diesem Faktor ab, und der Anteil des gesamten Variabilität erklären
mit diesem Faktor wird groß sein. Der zweite Faktor, erklärt der zweitgrößte Anteil der Variabilität unterliegen
der Einschränkung, dass der zweite Faktor orthogonal zur ersten ist, und so weiter für die nachfolgenden Faktoren. Sterben
entsprechenden Fraktionen entsprechend der ersten 50 Faktoren wurden für die Tests dargestellt, basierend auf Blum-
Blum-Shub-Sequenzen mit einer Länge 1.000.000. Dieser Graph zeigt, dass es keine große Redundanz bei
unsere Tests.

Die Korrelationsmatrix aus der z1, ... gebildet wurde zm über eine statistische Software-Anwendung aufgebaut
(SAS). Die gleiche Schlussfolgerung wurde durch die Struktur dieser Matrizen unterstützt. Der Grad der Vervielfältigung
unter den Tests scheint sehr gering zu sein.

Analyse

5. Bedienungsanleitung
Dieser Abschnitt beschreibt die Einrichtung und ordnungsgemäße Verwendung der statistischen Tests durch NIST entwickelt, die
in der NIST-Test-Code. Beschreibungen der Algorithmen und Datenstrukturen, die verwendet wurden sind
enthalten in diesem Abschnitt.

5.1 Über die Package

Diese Toolbox wurde speziell für Personen in leitenden statistischen Tests interessiert konzipiert
kryptographische (P) RNGs. Mehrere Implementierungen PRNGs während der Entwicklungsphase genutzt
das Projekt wurden ebenfalls berücksichtigt.

Dieses Paket wird das Problem der Bewertung (P) RNGs auf Zufälligkeit. Es wird nützlich sein, in:

- Identifizierung (P) RNGs die schwache (oder gemustert) Binärsequenzen produzieren,
- Entwicklung neuer (P) RNGs,
- Überprüfung, dass die Implementierungen von (P) RNGs richtig sind,
- Studium (P) RNGs in Normen beschrieben, und
- Untersuchung der Grad der Zufälligkeit von derzeit verwendeten (P) RNGs.

Die Ziele bei der Entwicklung des NIST statistischen Test Suite enthalten:

- Plattformunabhängigkeit: Der Quellcode wurde in ANSI C geschrieben jedoch einige Änderungen
  Möglicherweise müssen Sie gemacht, je nach Zielplattform und der Compiler sein.
- Flexibilität: Der Benutzer kann frei stellen ihre eigenen mathematischen Software-Routinen.
- Erweiterbarkeit: Neue statistische Tests können einfach integriert werden.

•

Vielseitigkeit: Die Test-Suite ist nützlich, in der Durchführung von Tests für PRNGs, RNGs und Verschlüsselungsalgorithmen.

•

Portabilität: Mit geringfügigen Modifikationen, Source-Code kann auf verschiedene Plattformen portiert werden. Sterben
NIST-Quellcode wurde mit einem Windows XP System mit Visual Studio 2005 Compiler und portiert
eine Ubuntu-Linux-System läuft gcc.

•

Orthogonalität: Eine vielfältige Reihe von Tests vorgesehen ist.

•

Effizienz: Linear Zeit oder Raum-Algorithmen wurden genutzt, wann immer möglich.

5.2 System-Anforderungen

Dieses Softwarepaket wurde ursprünglich auf einer SUN-Workstation unter dem Betriebssystem Solaris entwickelt
Systems. Alle Quellcode in ANSI C geschrieben wurde Die neuesten Versionen wurden auf einem Apple getan
MacBook Pro mit einem Intel Core 2 Duo-Prozessor mit dem gcc-Compiler.

In der Praxis können geringfügige Änderungen müssen bei der Portierung von Verfahren eingeführt werden, um sicherzustellen,
die korrekte Interpretation von Tests. Für den Fall, dass ein Benutzer zum Kompilieren und Ausführen des Codes auf einem Wunsch
andere Plattform, Sample-Daten und die entsprechenden Ergebnisse für jede der statistischen Tests wurden
zur Verfügung gestellt. Auf diese Weise wird der Anwender in der Lage sein, das Vertrauen, dass die Portierung statistische Test-Suite ist mein Gewinn ordnungsgemäß funktioniert. Für weitere Einzelheiten siehe Anhang B.

Für die Mehrzahl der statistischen Tests müssen Speicher dynamisch zugewiesen werden, um fortzufahren. Im

Bei diesem Arbeitsbereich nicht erbracht werden kann, liefert die statistischen Test eine diagnostische Meldung.

Analyse

5.3 Wie fange ich an

So richten Sie eine Kopie der NIST-Test-Code auf einer Workstation, folgen Sie den Anweisungen unten.

•

Kopieren Sie die sts.tar-Datei in das Root-Verzeichnis. Verwenden Sie die Anweisung, tar-xvf sts.tar, um
Entbündelung des Quellcodes.

•

Sechs Unterverzeichnisse und eine Datei sollte erstellt worden sein. Die Unterverzeichnisse sind:
data /, Experimente /, include / obj /, src / und templates /. Die Datei wird makefile.

•

Die Daten /-Unterverzeichnis ist für bereits bestehende RNG Dateien, die unter vorbehalten
Untersuchung. Derzeit sind zwei Formate werden unterstützt, dh Dateien, bestehend aus
ASCII Nullen und Einsen, und binäre Dateien.

•

Die Experimente /-Unterverzeichnis wird das Repository der empirischen Ergebnisse für die sein
statistischen Tests. Mehrere Unterverzeichnisse sollten darin enthalten sein. Dazu gehören
AlgorithmTesting /, BBS /, CCG /, G-SHA1 /, LCG /, MODEXP / MS /, QCG1 /,

QCG2 /, und XOR /. Alle aber die erste dieser Unterverzeichnisse soll das Geschäft Ergebnisse für die entsprechenden PRNG. Die AlgorithmTesting /-Unterverzeichnis ist die
Standard-Unterverzeichnis für empirische Ergebnisse entsprechend RNG Daten in die gespeicherten
Daten /-Unterverzeichnis.

- Die include / Unterverzeichnis enthält die Header-Dateien für die statistischen Tests, Pseudo-
Anzahl Generatoren, und die damit verbundenen Routinen.
- Der obj /-Unterverzeichnis enthält die Objekt-Dateien entsprechend der statistischen Tests
Pseudo-Zufallszahlen-Generatoren und andere zugehörige Routinen.
- Die src / Unterverzeichnis enthält die Quell-Codes für jede der statistischen Tests.
- Die Vorlagen /-Unterverzeichnis enthält eine Reihe von nicht-periodische Vorlagen für unterschiedliche
Blockgrößen, die durch die überlappenden Templates statistischen Test genutzt werden.
- Benutzer vorgegebenen Änderungen können in mehrere Dateien eingeführt werden. Dies wird
anschließend in Abschnitt 5.5.2 und Anhang A. diskutiert
- Bearbeiten Sie die makefile. Ändern Sie die folgenden Zeilen:
(A) CC (Ihr ANSI C-Compiler)
(B) ROOTDIR (das Root-Verzeichnis, das früher in den Prozess, zB vorgeschrieben war,
rng /)
-

Führen Sie nun makefile. Eine ausführbare Datei mit dem Namen zu beurteilen sollten in das Projekt erscheinen
Verzeichnis.

•

Die Daten können jetzt ausgewertet werden. Geben Sie den folgenden: Beurteilung <sequenceLength>, zB
Beurteilung 1000000 starten.
Folgen Sie der Menüführung.

5-2

Analyse

5,4 Data Input und Output der empirischen Ergebnisse
5.4.1 Data Input
Die Dateneingabe kann in eine von zwei Arten geliefert werden. Wenn der Benutzer über einen Stand-alone-Programm oder Hardware-Gerät
die implementiert einen RNG, kann der Benutzer möchte so viele Dateien von beliebiger Länge zu konstruieren, wie gewünscht.
Dateien sollten Binärsequenzen entweder als ASCII-Zeichen, bestehend aus Nullen und Einsen gespeichert werden, oder
als binäre Daten, wo jedes Byte besteht aus acht Bits im Wert von 0 und 1 ist. Die NIST-Statistische Test Suite
kann dann unabhängig untersuchen diese Dateien.

In dem Fall, dass Speicherplatz ein Problem ist, kann der Benutzer wollen die Referenz-Implementierung ändern
und Plug-in der Umsetzung der PRNG evaluiert. Die Bit-Streams werden direkt gespeichert werden
in der epsilon Datenstruktur enthält die binären Sequenzen.

5.4.2 Output der empirischen Ergebnisse
Die Ausgabe von Protokollen der empirischen Ergebnisse werden in zwei Dateien, Statistiken und Ergebnissen, die entsprechen gespeichert werden
jeweils auf die rechnerische Informationen zB Teststatistiken, Zwischen-Parameter und P-Werte
für jeden statistischen Test angewendet, um einen Datensatz.

Wenn diese Dateien nicht ordnungsgemäß erstellt werden, dann ist es sehr wahrscheinlich aufgrund der Unfähigkeit, um die Dateien für die offene Ausgabe. Siehe Anhang C für weitere Details.

5.4.3 Test Data Files
Six Probe-Dateien wurden erstellt und werden in die Daten /-Unterverzeichnis enthalten. Vier dieser Dateien
entsprechen den Mathematica9 erzeugten binären Ausbau mehrerer klassischen Zahlen über
1.000.000 Bits. Diese Dateien sind data.e, data.pi, data.sqrt2 und data.sqrt3. Der Mathematica-Programm
verwendet bei der Erstellung dieser Dateien im Anhang F. Ein Fünftel Datei data.sha1 gefunden werden kann, errichtet wurde unter Verwendung von
SHA-1 Hash-Funktion. Die endgültige Probe-Datei ist eine von einer vorgespannten PRNG produziert und soll nicht
bestimmte statistische Tests.

5,5 Programm-Layout
Die Test-Suite-Paket hat in einer Reihe von Modulen zerlegt worden, die gehören: statistische Tests
und (Pseudo-) Zufallszahlen-Generatoren, empirische Ergebnisse (hierarchische) Verzeichnisse und Daten.

Die drei Hauptkomponenten des NIST Testsuite sind die statistischen Tests, die

zugrunde liegenden mathematischen
Software und die Pseudo-Zufallszahlen-Generatoren untersucht. Weitere Komponenten sind die
Source-Code-Bibliothek-Dateien, die Daten-Verzeichnis und die hierarchische Verzeichnisstruktur (Experimente /) mit dem
Sample-Daten-Dateien und empirische Ergebnis-Protokolle beziehungsweise.

5.5.1 Allgemeines Programm
Die NIST-Test-Suite enthält fünfzehn Tests, die nützlich sein werden bei der Untersuchung und Auswertung der binären
Sequenzen, die durch zufällige und Pseudo-Zufallszahlen-Generatoren produziert. Wie in früheren Arbeiten auf diesem Gebiet,
statistischen Tests, die ersonnen werden müssen, unter einigen vermutet, Vertrieb, beschäftigen einen bestimmten Test
Statistik, wie die Anzahl der Durchläufe von Einsen oder die Anzahl, wie oft ein Muster erscheint in einem Bitstrom. Sterben
Mehrheit der Tests in der Testsuite entweder (1) untersuchen die Verteilung von Nullen und Einsen in einigen

9 Mathematica, Stephen Wolfram ist Computer-Algebra-System, http://www.wolfram.com/.

Analyse

Mode, (2) Studie der Oberschwingungen des Bitstrom Nutzung spektraler Methoden, oder (3) Versuch zu erkennen,
Muster über einige allgemeine Pattern-Matching-Technik auf der Grundlage der Wahrscheinlichkeitsrechnung oder Informationen

Theorie.

5.5.2 Globale Parameter
In der Praxis kann eine beliebige Anzahl von Problemen, wenn der Benutzer führt diese Software in unerforschte Bereiche.
Es ist plausibel, dass Sequenzlängen weit über das Testverfahren (dh in der Größenordnung von 106) kann
gewählt. Wenn der Speicher verfügbar ist, sollte es keinen Grund, warum die Software sollte nicht sein. Allerdings
in vielen Fällen sind die benutzerdefinierten Grenzwerte für Datenstrukturen und Arbeitsbereich vorgeschrieben. Unter diesen
Bedingungen, kann es notwendig sein, um bestimmte Parameter, wie die NUMOFTEMPLATES erhöhen.
Mehrere Parameter, die von einem Benutzer geändert werden können, sind in Tabelle 5-1 aufgeführt.

Tabelle 5-1 Benutzer vorgeschriebenen Statistische Test-Parameter

Source Code Parameter Default Parameter Beschreibung / Definition
ALPHA 0,01 Signifikanzniveau
MAXNUMOFTEMPLATES 40 Nicht-überlappende Vorlagen-Test
NUMOFTESTS 16 Max Anzahl der Tests
NUMOFGENERATORS 12 Max Anzahl der PRNGs

Der Parameter ALPHA bezeichnet das Signifikanzniveau, dass die Region von Akzeptanz und bestimmt
Ablehnung. NIST empfiehlt, dass ALPHA in den Bereich [0,001, 0,01] werden.

Der Parameter MAXNUMOFTEMPLATES gibt die maximale Anzahl von nicht-periodische Vorlagen,
kann durch die überlappende Template Matchings Test ausgeführt werden. Für Vorlagen der Größe m = 9, bis zu 148

möglichen nicht-periodische Vorlagen angewendet werden kann.

Die Parameter NUMOFTESTS und NUMOFGENERATORS, um die maximale Anzahl der Tests entsprechen
Das mag in der Test-Suite definiert werden, und die maximale Anzahl von Generatoren in der Testsuite angegeben,
jeweils.

5.5.3 Mathematische Software
Spezielle Funktionen durch die Test-Suite erforderlich sind die unvollständige Gamma-Funktion und der komplementären
Fehler-Funktion. Die kumulative Verteilungsfunktion, die auch als Standard normale Funktion bekannt ist, ist auch
erforderlich, kann aber im Hinblick auf die Fehler-Funktion ausgedrückt werden.

Einer der anfänglichen Bedenken in Bezug auf die Entwicklung der Referenz-Implementierung war die
Abhängigkeiten, die benötigt wurden, um zuverlässige mathematische Software für die speziellen Funktionen zu gewinnen
erforderlich, die statistischen Tests. Zur Lösung dieser Frage, macht die Test-Suite Verwendung der folgenden Bibliotheken:

Die Fast-Fourier-Transform-Routine wurde am http://www.netlib.org/fftpack/fft.c erhalten.
Die normale Funktion in diesem Code verwendet wurde im Hinblick auf die Fehler-Funktion zum Ausdruck gebracht.

Standard-Normal (kumulative Verteilung) Function
$

Analyse

Die komplementäre Fehlerfunktion (erfc) in dem Paket genutzt wird der
ANSI-C-Funktion in der enthaltenen
math.h Header-Datei und die zugehörige mathematische Bibliothek. Diese Bibliothek sollte während aufgenommen werden
Zusammenstellung.

Komplementäre Fehlerfunktion

Die unvollständige Gamma-Funktion ist eine Näherungsformel, deren Ursprung in der beschriebenen Basis
Handbook of Applied Mathematical Functions [1] und in den Numerical Recipes in C Buch [6].
Abhängig von den Werten der Parameter a und x, so kann die unvollständige Gamma-Funktion angenähert werden
entweder mit einem Kettenbruch Entwicklung oder eine Serienentwicklung.

Gamma-Funktion

! () Ztedtzt = unvollständige Gamma-Funktion

Paxaxaaetdttax (,)
wobei P (a, 0) = 0 und P (a, 8) = 1.

Unvollständige Gamma-Funktion

QaxPaxaxaaetdttax (,)(,)
wo Q (a, 0) = 1 und Q (a, 8) = 0.

NIST hat sich entschieden, die Cephes Programmiersprache C spezielle Funktionen

Mathematik-Bibliothek in die Test-Software verwenden.
Cerphes kann bei http://www.moshier.net/-Cephes oder auf der GAMS-Server finden Sie unter
http://gams.nist.gov/serve.cgi/Package/CEPHES/. Die spezifischen Funktionen, die genutzt werden igamc (für
der komplementären unvollständige Gamma-Funktion) und lgam (für die logarithmische Gamma-Funktion).

5,6 Ausführen der Test Code
Ein Beispiel NIST Statistische Test Suite monolog wird nachfolgend beschrieben.
Hinweis: In diesem Abschnitt fett Artikel
zeigen Eingang.

Um den NIST statistische Test-Suite aufrufen, beurteilen Typ, gefolgt von der gewünschten Bitstrom Länge n.
Zum Beispiel, zu beurteilen 100000. Eine Reihe von Menüführung wird angezeigt, um die Daten auswählen, um sein
analysiert und die statistischen Tests angewendet werden. Der erste Bildschirm, sieht wie folgt aus:

Analyse

Generator-Optionen

[00] Input File [01] linearen Kongruenz
[02] Quadratische Kongruenz I [03] Quadratische Kongruenz II
[04] Cubic Kongruenz [05] XOR

[06] modulare Exponentiation [07] Blum-Blum-Shub
[08] Micali-Schnorr [09] G mit SHA-1
OPTION ----> 0
Benutzer vorgeschriebenen Input File: data / data.pi

Sobald der Benutzer einen bestimmten Datensatz oder PRNG verschrieben hat, müssen die statistischen Tests angewandt werden
Ausgewählt wurde. Der folgende Bildschirm wird angezeigt:

Statistische Tests

[01] Frequenz [02] Block Frequency
[03] Summenwerte [04] Runs
[05] längsten Abfahrten des Ones [06] Rang
[07] Spectral-Discrete Fourier Transform [08] nichtperiodische Template Matchings
[09] Overlapping Template Matchings [10] Universal Statistical
[11] Ungefähre Entropy [12] Random Excursions
[13] Random Excursions Variant [14] Serielle
[15] lineare Komplexität
ANLEITUNG
Geben Sie 0, wenn Sie nicht möchten, dass alle gelten
statistischen Tests zu jeder Sequenz und 1, wenn Sie tun.

Geben Sie Choice: 0

In diesem Fall hat 0 ausgewählt worden, um Interesse an der Anwendung eine Teilmenge der verfügbaren statistischen Tests zeigen.
Der folgende Bildschirm wird dann angezeigt.

ANLEITUNG
Geben Sie eine 0 oder 1, um anzugeben, ob die nummerierten

statistischen Tests sollten jede Sequenz angewendet werden. Für Beispielsweise gilt 111111111111111 jeden Test zu jeder Sequenz.

123456789111111
012345

000000001000000

5-6

Analyse

Wie oben gezeigt, war der einzige Test angewendet Nummer 9, die überlappende Vorlagen zu testen. Eine Abfrage für die gewünschte Stichprobengröße wird dann gemacht.

Wie viele Bitströme erzeugt werden soll? 10

Zehn Sequenzen werden analysiert mit dem data.pi-Datei sein. Da eine Datei war, da die Daten Spezifikation ausgewählt Modus wird eine nachfolgende Abfrage gemacht in Bezug auf die Darstellung der Daten. Der Benutzer muss angeben, ob die Datei besteht aus Bits im ASCII-Format oder hexadezimale Strings in binären Format gespeichert sind.

Input File Format:

[0] ASCII - Eine Folge von ASCII 0 und 1 ist
[1] Binary - Jedes Byte in Datei enthält 8 Datenbits

Wählen Sie den Eingabemodus: 0
Da die Daten aus einer langen Abfolge von Nullen und Einsen, war 0 gewählt. Da alle notwendigen Ein-
Parameter der Test-Suite geht um die Sequenzen zu analysieren.

Statistische Tests In Progress .....

Sobald die Prüfung abgeschlossen ist, können die empirischen Ergebnisse in den Experimenten /-Unterverzeichnis gefunden werden.

Statistische Tests abgeschlossen !!!!!!!!!!!

Nach der Fertigstellung wird eine eingehende Analyse durchgeführt, um die Analyse der empirischen Ergebnisse zu vereinfachen. Zwei
Arten von Analysen durchgeführt werden. Eine Art untersucht den Anteil von Sequenzen, die eine statistische Pass
zu testen. Die andere Art untersucht die Verteilung der P-Werte für die einzelnen statistischen Tests. Weitere Einzelheiten sind
geliefert im folgenden Abschnitt.

5,7 Interpretation der Ergebnisse
Eine analytische Routine wurde hinzugefügt, um die Interpretation der Ergebnisse zu erleichtern. Eine Datei
finalAnalysisReport wird generiert, wenn statistische Tests abgeschlossen sind. Der Bericht enthält eine Zusammenfassung der
empirische Ergebnisse. Die Ergebnisse werden über eine Tabelle mit p Zeilen und q Spalten dargestellt. Die Zahl der
Reihen, p, entspricht der Anzahl von statistischen Tests angewandt. Die Anzahl der Spalten, q = 13,
verteilten sich wie folgt: Spalten 1-10, um die Frequenz der P-values10 entsprechen, ist Spalte 11 der P-Wert

, die entsteht, über die Anwendung eines Chi-Quadrat-test11 ist Spalte 12 der Anteil der binären Sequenzen,
vergangen, und der 13. Spalte wird die entsprechende statistische Test. Ein Beispiel ist in Abbildung 5-1 gezeigt.

10 Die Einheit Intervall ist in zehn diskrete Behälter aufgeteilt.
11 I n, um die Einheitlichkeit der P-Werte in der i-ten statistischen Test zu bewerten.

Analyse

ERGEBNISSE für die Einheitlichkeit von P-Werten und der Anteil der PASSING SEQUENCES

C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 P-VALUE ANTEIL statistischen Test

6 12 9 12 8 7 8 12 15 11 0,616305 0,9900 Frequency

11 11 12 6 10 9 8 9 17 7 0,474986 0,9900 Cusum
6 10 8 14 16 10 10 6 5 15 0,129620 0,9900 Cusum
7 9 9 11 11 11 8 12 12 10 0,978072 0,9900 Serielle

13 6 13 15 9 7 3 11 13 10 0,171867 0,9600 Serielle

Die minimale Erfolgsquote für jeden statistischen Test mit Ausnahme der zufälligen Exkursion (Variante)-Test ist ungefähr = 0,960150 für eine Probe
size = 100 binäre Sequenzen.

Für weitere Richtlinien zu konstruieren einer Wahrscheinlichkeit Tabelle mit dem MAPLE-Programm
sofern in den Nachtrag Abschnitt der Dokumentation.

Abbildung 5-1 Darstellung der Final Analysis Report

Analyse

Anhang A Source Code
Hierarchische Verzeichnisstruktur

M. /
Beurteilung der NIST Statistische Test Suite ausführbare Datei aufgerufen wird, zu beurteilen. Diese Datei wird
erzeugt bei der Ausführung der mitgelieferten Makefile.

makefile
Die NIST-Statistische Test Suite makefile. Diese Datei wird aufgerufen, um zu bauen die gesamte Test-Suite und produzieren die ausführbare Datei zu beurteilen.

Daten /

Dieses Unterverzeichnis enthält Beispieldaten Dateien analysiert werden.
2Diese sind die
3

binären Erweiterungen bekannt Konstanten wie e, p,, und.

Experimente / Dieses Unterverzeichnis enthält das empirische Ergebnis
Unterverzeichnisse für jeden RNG.

AlgorithmTesting / BBS /
CCG / G-SHA-1 /
LCG / MODEXP /
MS / QCG1 /
QCG2 / XOR /

Für jedes Unterverzeichnis gibt es eine Reihe von verschachtelten Verzeichnissen, das heißt,

ApproximateEntropy / BlockFrequency /
CumulativeSums / FFT /
Frequenz / LinearComplexity /
LongestRun / NonOverlappingTemplates /
OverlappingTemplates / RandomExcursions /
RandomExcursionsVariant / Rank /
Läuft / Serial /
Universal /

Für jedes verschachteltes Verzeichnis gibt es zwei Dateien nach dem Ausführen eines erstellten

einzelnen statistischen Tests. Die Ergebnisse Datei enthält einen P-Wert-Liste für jeden binären
Sequenz, und die Statistik-Datei enthält eine Liste der statistischen Daten für jede binäre Folge.

include /
Dieses Unterverzeichnis enthält alle Header-Dateien, dass jede globale verschreiben Variablen, Konstanten und Datenstrukturen in die Referenz-Implementierung genutzt. Darüber hinaus enthält das Unterverzeichnis alle Funktions-Deklarationen und Prototypen.

cephes.h config.h
decls.h defs.h
externs.h generators.h
genutils.h matrix.h
stat_fncs.h utilities.h

obj /
Dieses Unterverzeichnis enthält die Objekt-Dateien entsprechend der Quellcode-Dateien.
Diese Dateien werden bei der Ausführung der mitgelieferten Makefile generiert werden.

Analyse

src /
Dieses Unterverzeichnis enthält die Quell-Codes für die statistischen Tests.

approximateEntropy.c: Ungefähre Entropy-Test
assess.c: Der Treiber für dieses Paket
blockFrequency.c: Block-Frequenz-Test
cephes.c: Definiert die unvollständige Gamma-Funktion
cusum.c Summenwerte-Test
dfft.c: Die diskrete Fourier-Transformation Routine
discreteFourierTransform.c

: Spectral-Test
frequency.c: Frequenz-Test
generators.c: Source-Code für das eingebaute Generatoren
genutils.c: Support-Routinen für die Generatoren
linearComplexity.c: Linear Complexity-Test
longestRunOfOnes.c: Längste Abfahrt von Einsen-Test
matrix.c: Source-Code für die Bestimmung des Ranges

für binäre Matrizen
nonOverlappingTemplateMatchings.c
: Nicht überlappende Template Matchings-Test
OverlappingTemplateMatchings.c

: Überlappende Template Matchings-Test
randomExcursions.c: Random Excursions-Test
randomExcursionsVariant.c

: Random Excursions Variant-Test
rank.c: Rank-Test
runs.c: Testläufe
serial.c: Serial-Test
universal.c: Universal Test
utilities.c: Utility-Funktionen ...

templates /
Dieses Unterverzeichnis enthält die Vorlagen (oder Muster), die in die ausgewertet werden
Überlappenden Template Matching Test. Die entsprechende Datei wird geöffnet
die vorgeschriebene Vorlage Blocklänge m. Derzeit ist die einzige Option für die nichtperiodische Vorlagen gespeichert wurden sind diejenigen, die in [2,21] liegen. Für Den Fall,
dass m> 21, muss der Benutzer vor Berechnung der nicht-periodische Vorlagen.

template2 template3 template4 template5 template6
template7 template8 template9 template10 template11
template12 template13 template14 template15 template16
template17 template18 template19 template20 template21

Analyse

Informationen zur Konfiguration von

Die folgenden Parameter sind in der Datei defs.h.

1. # Define ALPHA 0,01
2. # Define MAXNUMOFTEMPLATES 148
3. # Define NUMOFTESTS 15
4. # Define NUMOFGENERATORS 10
5. # Define MAXFILESPERMITTEDFORPARTITION 148

Zeilen 1-5 entsprechen suite Parameter, die voreingestellt sind zu testen. Unter verschiedenen Bedingungen, der Benutzer
kann beschließen, sie zu ändern.

Zeile 1 bezieht sich auf das Signifikanzniveau. Es wird empfohlen, dass der Benutzer das Niveau im Bereich wählen
[0.001,0.01].

Line 2 bezieht sich auf die maximale Anzahl der Vorlagen, die in der überlappende Template verwendet werden kann,
Passende zu testen. Der Wert 148 entspricht der Anzahl der Vorlagen der Länge 9.

Line 3 bezieht sich auf die maximale Anzahl der Tests, die in der Test-Suite unterstützt wird. Wenn der Benutzer fügt
zusätzliche Tests, sollte dieser Parameter erhöht werden.

Linie 4 bezieht sich auf die maximale Anzahl von Generatoren, die in dem Paket unterstützt wird. Wenn der Benutzer fügt
zusätzliche Generatoren, sollte dieser Parameter erhöht werden.

Zeile 5 bezieht sich auf die maximale Anzahl von Dateien, die von der partitionResultFile zerlegt werden kann
Routine. Diese Routine ist nur für bestimmte Tests, bei denen mehr als ein P-Wert je produziert wird angewendet
Sequenz. Diese Routine zersetzt die entsprechenden Ergebnisse Datei in einzelne Dateien, data001, data002,
data003, ...

Analyse

Anhang B Empirische Ergebnisse für Sample Data

Der Benutzer wird aufgefordert zu überprüfen, ob die statistischen Test-Suite einwandfrei funktioniert. Aus diesem Grund, fünf
Beispiel-Dateien bereitgestellt wurden. Diese fünf Dateien sind: (1) data.pi, (2) data.e, (3) data.sha1, (4)
data.sqrt2, und (5) data.sqrt3. Für jede Datei, die alle die statistischen Tests angewendet und die Ergebnisse
aufgezeichnet in den folgenden Tabellen. The Block Frequency, nicht überlappende Template Matching,
Overlapping Template Matching, Ungefähre Entropy, Serial und lineare Komplexität Tests des Benutzers erforderlich
vorgeschriebenen Eingabeparameter. Die genauen Werte in diesen Beispielen verwendet haben in Klammern aufgenommen
neben dem Namen des statistischen Tests. Im Falle des zufälligen Ausflüge und Exkursionen random
Variante Tests nur eine der möglichen 8 und 18 P-Werte jeweils berichtet wurde. Jede Probe wird
1.000.000 Bit Länge.

Beispiel # 1: Die binäre Ausbau der p

Statistische Test-P-Wert
Frequency 0.578211
Block Frequenz (m = 128) 0,380615
Cusum-Forward 0.628308
Cusum-Reverse 0.663369
Läuft 0.419268
Longruns Ones 0.024390
Rang 0.083553
Spectral DFT 0.012947
Nicht überlappende Templates (m = 9, B = 000000001) 0,165757
Überlappende Templates (m = 9) 0,296897

Universal-0.669012
Ungefähre Entropy (m = 10) 0,361595
Zufällige Ausflüge (x = +1) 0,844143
Zufällige Ausflüge Variant (x = -1) 0,760966
Lineare Komplexität (M = 500) 0,255475
Serial (m = 16,) 0,143005

B-1

Analyse

Beispiel # 2: Die binäre Ausbau der e

Statistische Test-P-Wert
Frequency 0.953749
Block Frequenz (m = 128) 0,211072
Cusum-Forward 0.669887
Cusum-Reverse 0.724266
Läuft 0.561917
Longruns Ones 0.718945
Rang 0.306156
Spectral DFT 0.443864
Überlappenden Templates (m = 9, B = 000000001) 0,078790
Überlappende Templates (m = 9) 0,110434
Universal-0.282568
Ungefähre Entropy (m = 10) 0,700073
Zufällige Ausflüge (x = +1) 0,778616
Zufällige Ausflüge Variant (x = -1) 0,826009
Lineare Komplexität (M = 500) 0,826335
Serial (m = 16,) 0,766182

Beispiel # 3: Ein G-SHA-1 binäre Folge

Statistische Test-P-Wert
Frequency 0.604458
Block Frequenz (m = 128) 0,091517
Cusum-Forward 0.451231
Cusum-Reverse 0.550134
Läuft 0.309757
Longruns Ones 0.657812
Rang 0.577829
Spectral DFT 0.086702
Überlappenden Templates (m = 9, B = 000000001) 0,496601
Überlappende Templates (m = 9) 0,339426
Universal-0.411079
Ungefähre Entropy (m = 10) 0,982885
Zufällige Ausflüge (x = +1) 0,000000
Zufällige Ausflüge Variant (x = -1) 0,000000
Lineare Komplexität (M = 500) 0,309412
Serial (m = 16,) 0,760793

Analyse

Beispiel # 4: Die Binär-Erweiterung

2
Statistische Test-P-Wert
Frequency 0.811881

Block Frequenz (m = 128) 0,833222
Cusum-Forward 0.879009
Cusum-Reverse 0.957206
Läuft 0.313427
Longruns Ones 0.012117
Rang 0.823810
Spectral DFT 0.267174
Überlappenden Templates (m = 9, B = 000000001) 0,569461
Überlappende Templates (m = 9) 0,791982
Universal-0.130805
Ungefähre Entropy (m = 10) 0,884740
Zufällige Ausflüge (x = +1) 0,216235
Zufällige Ausflüge Variant (x = -1) 0,566118
Lineare Komplexität (M = 500) 0,317127
Serial (m = 16,) 0,861925

Beispiel # 5: Die Binär-Erweiterung

3
Statistische Test-P-Wert
Frequency 0.610051
Block Frequenz (m = 128) 0,473961
Cusum-Forward 0.917121
Cusum-Reverse 0.689519
Läuft 0.261123
Longruns Ones 0.446726
Rang 0.314498
Spectral DFT 0.463412
Überlappenden Templates (m = 9, B = 000000001) 0,532235
Überlappende Templates (m = 9) 0,082716
Universal-0.165981
Ungefähre Entropy (m = 10) 0,180481

Zufällige Ausflüge (x = +1) 0,783283
Zufällige Ausflüge Variant (x = -1) 0,155066
Lineare Komplexität (M = 500) 0,346469
Serial (m = 16,) 0,157500

B-3

Analyse

Anhang C Erweiterung der Test-Suite
Einbindung zusätzlicher Statistische Tests

Um zu einem anderen statistischen Test auf die Test-Suite hinzugefügt, sollte der Anwender folgende Änderungen vornehmen:

1.
[In der Datei include / defs.h]
Legen Sie einen Test Input-Parameter in die _testParameters Struktur. Erhöhen Sie den Wert von
NUMOFTESTS durch die Anzahl der Tests hinzugefügt werden. Fügen Sie einen für den neuen Test zu definieren, zum Beispiel:

Define TEST_MYNEWTEST 16

2.
[In der Datei include / decls.h]
Setzen Sie die neue Test-Namen, zB "MyNewTest", in die testNames Array.
3.
[In der Datei include / stat_fncs.h]

Legen Sie die statistischen Test Funktionsprototyp Erklärung.
4.
[In der Datei src / myNewTest.c]
Definieren Sie den statistischen Test-Funktion. Hinweis: Der Programmierer sollte fprintf Aussagen einbetten
mit stats [TEST_MYNEWTEST], und die Ergebnisse [TEST_MYNEWTEST] als der entsprechende Ausgang
Kanal für das Schreiben von Zwischen-Teststatistik Parameter und P-Werte jeweils.

5.
[In der Datei src / utilities.c]
(A)
In der Funktion chooseTests, ändern Sie die printf die Angaben korrekt angezeigt werden der neue Test zu nennen. Für
Ändern Sie beispielsweise
printf ("[15] lineare Komplexität \ n \ n");

zu

printf ("[15] lineare Komplexität [16] MyNewTest \ n \ n");

und ändern Sie die folgenden Zeilen Code (wie von der tatsächlichen Anzahl der insgesamt Tests modifiziert):

printf ("1234567891111111 \ n");
printf ("0123456 \ n");

Hinweis: Für jede PRNG in den Experimenten / Verzeichnis definiert, ein Unterverzeichnis myNewTest muss

werden geschaffen.

(C) Wenn ein Eingang-Test-Parameter ist erforderlich, in der Funktion, fixParameters, legen Sie die folgenden Zeilen
Code (unter der Annahme, dass myNewTestParameter eine ganze Zahl ist). Zum Beispiel,
einfügen

```
if (testVector [TEST_MYNEWTEST] == 1) {
counter + +;
```

C-1

Analyse

```
if (counter == TestID) {
printf ("Geben Sie MyNewTest Parameter Wert:");
scanf ("% d", & tp.myNewTestParameter);
printf ("\ n");
fortzusetzen;


}
}
```

(D) Embed den Test-Funktion aufrufen in die nist_test_suite Funktion. Zum Beispiel:

```
if ((testVector [0] == 1) | | (testVector [TEST_MYNEWTEST] == 1))
myNewTest (tp.myNewTestInputParameters, tp.n);
```

Einbindung zusätzlicher PRNGs

Um eine PRNG die Testsuite hinzuzufügen, sollte der Anwender folgende Änderungen vornehmen:

1.
[In der Datei include / defs.h]
Erhöhen Sie die variable NUMOFGENERATORS nach dem anderen.
2.
[In der Datei include / decls.h]
Setzen Sie die neue PRNG Namen in das generatorDir array, zum Beispiel
char generatorDir [20] [20] = {"AlgorithmTesting"
..., "G-SHA1", "MYNEWPRNG"};

3.
[In der Datei include / generators.h]
Legen Sie die Generator-Funktion Prototyp Erklärung. Zum Beispiel,
void myNewPRNG ();

4.
[In der Datei-Generatoren / generators.c]
Definieren Sie die Generator-Funktion. Das allgemeine Schema für jeden PRNG im Test definiert
Suite ist wie folgt:

Speicherplatz für epsilon, die n-Bit-Bitfeld-Arrays.

für i = 1 bis numOfBitStreams {

Konstruieren Sie einen n-Bit-Sequenz mit myNewGenerator und speichern
in epsilon.
Rufen Sie die nist_test_suite.

}
Deallocate den Raum gegeben, um epsilon.

Hinweis: Ein Unterverzeichnis namens myNewPRNG / muss in den Experimenten / Verzeichnis erstellt werden.
Unter diesem neuen Verzeichnis, müssen eine Reihe von Unterverzeichnissen für jedes der Test-Suite erstellt werden
statistischen Tests. Das Skript createScript aufgenommen wurde, um diesen Vorgang zu erleichtern.

C-2

Analyse

5. [In der Datei src / utilities.c]
(A) In der Funktion displayGeneratorOptions, Einschub folgende Code-Zeile:
printf ("[10] MyNewPRNG \ n \ n");

(B) In der Funktion generatorOptions, legen Sie die folgenden Codezeilen:
Fall 10:
* StreamFile = "myNewPRNG";
break;

(C) In der Funktion invokeTestSuite, legen Sie die folgenden Codezeilen:

Fall 10:

myNewPRNG ();

break;

Analyse

Anhang D Beschreibung des Reference Pseudo-Zufallszahlengeneratoren
Die NIST-Statistische Test Suite liefert dem Anwender mit neun Pseudo-Zufallszahlen-Generatoren. Eine kurze Beschreibung der einzelnen Pseudo-Zufallszahlen-Generator folgt. Der Benutzer geliefert Sequenzlänge bestimmt die Anzahl der Iterationen für jeden Generator.

D.1 linearen Kongruenz-Generator (LCG)
Die Eingabeparameter für die Fishman und Moore12 LCG13 ist im Code festgelegt, sondern kann vom Benutzer geändert werden.
Input Parameter:
z0
= 23482349
Beschreibung:
Angesichts eines Samens z0, sind die nachfolgenden Zahlen basieren auf zi +1 berechnet = a * zi mod (231-1), wo eine ist eine Funktion

von den aktuellen Zustand. Diese Zahlen werden dann auf einheitliche Werte in [0,1] umgewandelt. Bei jedem Schritt, Ausgang '0 ' wenn die Zahl =
0,5, sonst Ausgabe '1 '.

D.2 Quadratische Kongruenz-Generator I (QCG-I)
Die Eingangsparameter der QCG-I werden in Code festgelegt, sondern kann vom Benutzer geändert.
Eingangsparameter:
p =
987b6a6bf2c56a97291c445409920032499f9ee7ad128301b5d0254aa1a9633fdbd378

d40149f1e23a13849f3d45992f5c4c6b7104099bc301f6005f9d8115e1
x0 =
3844506a9456c564b8b8538e0cc15aff46c95e69600f084f0657c2401b3c244734b62e
a9bb95be4923b9b7e84eeaf1a224894ef0328d44bc3eb3e983644da3f5

Beschreibung:
Mit einem 512-Bit-Primzahl p und eine zufällige 512-Bit-Seed-x0, konstruieren

nachfolgenden Elemente (jeweils 512-Bit-
Zahlen) in der Reihenfolge über die Regel:
xi +1 = xi2 mod p für i =
0 ist.

D.3 Quadratische Kongruenz-Generator II (QCG-II)
Die Input-Parameter, um die QCG-II ist im Code festgelegt, sondern kann vom Benutzer geändert werden.

12 Fishman, G. S. und L. R. Moore (1986). Eine ausführliche Analyse der multiplikativen Kongruenz Zufallszahlengeneratoren mit
Modul 2 ** 31-1, SIAM Journal on Scientific and Statistical Computation, 7, 24-45.

13 Zusätzliche Informationen können in Kapitel 16 (Pseudo-Random Sequence Generatoren & Stream Ciphers), Abschnitt 16.1 gefunden werden
(Linear-kongruente Generatoren) von Bruce Schneier Buch Applied Cryptography: Protocols, Algorithms and Source
Code in C, 2. Auflage, John Wiley & Sons, 1996.

Analyse

Input Parameter:

x0 =
7844506a9456c564b8b8538e0cc15aff46c95e69600f084f0657c2401b3c244734b62e
a9bb95be4923b9b7e84eeaf1a224894ef0328d44bc3eb3e983644da3f5
Beschreibung:
Mit einem 512-Bit-Modul und eine zufällige 512-Bit-Seed-x0, konstruieren nachfolgenden Elemente (jeweils 512-Bit-

Zahlen) in der Reihenfolge über die Regel:
xi +1 = 2xi2 + 3xi + 1 mod 2512, für i =
0 ist.

D.4 Cubic Kongruenz Generator II (CCG)
Die Input-Parameter, um die CCG ist im Code festgelegt, sondern kann vom Benutzer geändert werden.
Input Parameter:
x0 =
7844506a9456c564b8b8538e0cc15aff46c95e69600f084f0657c2401b3c244734b62ea

9bb95be4923b9b7e84eeaf1a224894ef0328d44bc3eb3e983644da3f5
Beschreibung:

Bei einer 512-Bit-Seed-x0, konstruieren anschließenden 512-Bit-Strings über die Regel:

xi +1 = xi3 mod 2512, für i =
0 ist.

D.5 Exklusiv-ODER-Generator (Xorg)
Die Input-Parameter, die Xorg ist eine 127-Bit-Samen, die im Code festgelegt ist, sondern können vom Benutzer modifiziert werden.
Input Parameter:

xxx12127, = 000101101101100100010111100100101001101110110100010000000101

01111111010100100001010110110000000000100110000101110011111111100111
Beschreibung:

Wählen Sie eine Bitfolge,

xxx12127,,, K. Construct nachfolgenden Bits über die Regel: XXXIII = 1127, für i = 128.
Modulare Exponentiation Generator (MODEXPG)

Die Eingangsparameter der MODEXPG sind im Code behoben, aber sie können vom Benutzer modifiziert werden.
Eingangsparameter:
seed = 7AB36982CE1ADF832019CDFEB2393CABDF0214EC

Analyse

p =
987b6a6bf2c56a97291c445409920032499f9ee7ad128301b5d0254aa1a9633fdbd378
d40149f1e23a13849f3d45992f5c4c6b7104099bc301f6005f9d8115e1

g =
3844506a9456c564b8b8538e0cc15aff46c95e69600f084f0657c2401b3c244734b62ea
9bb95be4923b9b7e84eeaf1a224894ef0328d44bc3eb3e983644da3f5

Beschreibung:

Eine Folge {xi} von 512-Bit-Pseudo-Zufallszahlen lassen sich wie folgt generiert werden:

Wählen Sie eine 512-Bit-Primzahl p und eine Basis g, wie in den Digital Signature Standard (DSS).
gpyimod,
Wählen Sie eine beliebige
160-Bit-Seed y. Lassen x1 = gseed mod p und xi +1 =

für i =
1, wo yi ist die niedrigste Ordnung 160 Bit

von xi. Splicing zusammen die {xi} erzeugt einen n-Bit-Sequenz.

D.7 Secure Hash Generator (G-SHA1)
Die Input-Parameter, um die G-SHA1 sind in Code festgelegt, sondern können vom Benutzer modifiziert werden. Die Länge des Schlüssels,
KEYLEN im Intervall [160, 512] gewählt werden.

Eingangsparameter:
seedlen = 160
Xseed = 237c5f791c2cfe47bfb16d2d54a0d60665b20904

KEYLEN = 160
Xkey = ec822a619d6ed5d9492218a7a4c5b15d57c61601

Beschreibung:
Für eine detaillierte Beschreibung der G-SHA1 (die FIPS 186 Einweg-Funktion mit SHA-1), besuchen Sie
http://www.cacr.math.uwaterloo.ca/hac/about/chap5.pdf, insbesondere S. 175.

D.8 Blum-Blum-Shub (BBSG)
Die Eingangsparameter der BBSG sind nicht im Code behoben. Sie sind variable Parameter, die Zeit sind
abhängig ist. Die drei erforderlichen Parameter sind zwei Primzahlen p und q, und eine Zufallszahl s.

Eingangsparameter:

Zwei Primzahlen p und q, so dass jedes kongruent 3 modulo 4 ist. Eine Zufallszahl s (die Saat), gewählt in
das Intervall [1, pq-1], so dass ggT (s, pq) = 1. Die Parameter p, q und s sind nicht im Code behoben; damit die
Benutzer nicht in der Lage sein, die ursprüngliche Reihenfolge zu rekonstruieren, weil diese Werte variieren (dh sie sind
abhängig von der Systemzeit). Die Input-Parameter wurden so befestigt, der Code die gleichen reproduzieren
Resultate.

Beschreibung:

Für eine detaillierte Beschreibung des Blum-Blum-Shub
Pseudo-Zufallszahlen-Generator finden Sie unter
http://www.cacr.math.uwaterloo.ca/hac/about/chap5.pdf, insbesondere S. 186.

D-3

Analyse

Micali-Schnorr-Generator (MSG)

Die Input-Parameter, um die MSG nicht im Code behoben. Sie sind variable Parameter, die Zeit sind
abhängig ist. Die vier erforderlichen Parameter sind zwei Primzahlen p und q eine ganze Zahl e, und der Same x0.

Eingangsparameter:

Zwei Primzahlen p und q. Ein Parameter e, so gewählt, dass 1 <e <f
= (P-1) (q-1), gcd (e, f) = 1 und 80e <N
= Floor (lg n + 1). Ein zufälliger Reihenfolge x0 (die Saat), bestehend aus r (eine Funktion von e und n) Bits gewählt wird.
Die Parameter e, p, q, und x0 nicht im Code behoben; damit der Benutzer nicht in der Lage, zu rekonstruieren
ursprüngliche Reihenfolge, weil diese Werte variieren (dh sie sind abhängig von der Systemzeit). Der Eingang
Parameter wurden so befestigt, der Code die gleichen Ergebnisse zu reproduzieren wird.

Beschreibung:

Für eine detaillierte Beschreibung der Micali-Schnorr von
Pseudo-Zufallszahlen-Generator finden Sie unter
http://www.cacr.math.uwaterloo.ca/hac/about/chap5.pdf, insbesondere S. 186.

D-4

Analyse

D.10 Testergebnisse
Die folgende Tabelle zeigt Test-by-Test Ausfälle für die oben genannten Referenz-Generatoren.

Statistische Tests ExcessiveR
ejections
Fehlt
Uniformity Generator
Frequency X X modulare Exponentiation
X X Cubic Kongruenz
X X Quadratische Kongruenz (Typ I)
Block Frequency X Cubic Kongruenz
X X XOR
Cusum X X Micali-Schnorr
X X modulare Exponentiation
X X Cubic Kongruenz
X X Quadratische Kongruenz (Typ I)
Läuft X modulare Exponentiation
X X Cubic Kongruenz
X Quadratische Kongruenz (Typ I)
Rang X X XOR
Spectral X X Cubic Kongruenz
X Quadratische Kongruenz (Typ II)
Aperiodische Templates X ANSI X9.17
X Micali-Schnorr
X modulare Exponentiation
X X Cubic Kongruenz
X Quadratische Kongruenz (Typ I)
X Quadratische Kongruenz (Typ II)
X X XOR
Periodische Vorlagen X modulare Exponentiation
X X XOR
Annähernd
Entropie
X X modulare Exponentiation
X X Cubic Kongruenz
X X Quadratische Kongruenz (Typ I)
X X XOR
Serielle X X modulare Exponentiation
X X Cubic Kongruenz
X X Quadratische Kongruenz (Typ I)
X X XOR

Tabelle D.1: Illustration der Ablehnung / Uniformity Failures

Analyse

Anhang E Numerical Algorithm Issues

Für jede binäre Folge, muss eine Person statistischen Test zu produzieren mindestens ein P-Wert. P-Werte sind
basierend auf der Auswertung von speziellen Funktionen, die so genau wie möglich auf der Zielplattform muss.
Die Log-Dateien von jedem statistischen Test Bericht P-Werte mit sechs Ziffern der Präzision gefertigt, die sollten
ausreichend. Allerdings, wenn größere Genauigkeit gewünscht ist, ändern Sie die printf-Anweisungen in jeder statistischen Test
dementsprechend.

Während der Testphase, NIST allgemein Sequenzen in der Größenordnung 106 ausgewertet, daher Ergebnisse basieren
auf dieser Annahme. Möchte der Anwender zu längeren Sequenz Längen wählen, dann beachten Sie, dass numerische
Berechnungen können ungenau sein wegen Maschinen-oder algorithmische Einschränkungen. Für weitere Informationen über
numerischen Analyse Angelegenheiten, siehe [6] 14.

Für die Zwecke der Veranschaulichung sind Beispiel-Parameter-Werte und die entsprechenden speziellen Funktionswerte
in Tabelle E.1 und Tabelle E.2. Tabelle E.1 vergleicht die Ergebnisse für die unvollständige Gamma-Funktion für
ausgewählte Parameter Werte für a und x. Die Ergebnisse sind für Maple15, Matlab10 gezeigt, und die Numerische
Recipe16 Routinen. Daran erinnern, dass die Definitionen für die Gamma-Funktion und die unvollständige Gamma-Funktion
definiert sind, bzw., wie:

! () Ztedtzt =),(,
wo Q (a, 0) = 1 und Q (a, 8) = 0.

Da der Algorithmus in die Test-Suite Umsetzung der unvollständige Gamma-Funktion verwendet wird, basiert auf
die numerische Rezept-Codes ist es offensichtlich, dass die Funktion genau zu mindestens der siebten Dezimalstelle.
Für große Werte von a, wird die Genauigkeit beeinträchtigen, wie das Vertrauen in das Ergebnis (es sei denn, ein Computer
Algebra-System wird eingesetzt, um hohe Präzision Berechnungen zu gewährleisten).

Tabelle E.2 vergleicht die Ergebnisse für die komplementäre Fehlerfunktion (siehe Abschnitt 5.3.3) für ausgewählte
Parameterwerte für x. Die Ergebnisse sind für ANSI C, Maple und Matlab dargestellt. Daran erinnern, dass die Definition
für die komplementäre Fehlerfunktion ist:

14 Besuchen Sie http://www.nr.com/, insbesondere Abschnitt 1.1 (Error, Genauigkeit und Stabilität).

15 Siehe Abschnitt 1.2, Begriffe und Abkürzungen.

16 Die Parameterwerte für eps und itmax wurden 3x10-15 und 2.000.000 bzw. fixiert.
Sonderfunktion Routinen auf
Numerische Rezept-Codes werden von nicht-proprietären Codes in naher Zukunft ersetzt werden.

E-1

Analyse

a = x = 600 Q (a, x) a = x = 800 Q (a, x)
Ahorn
Matlab
Test Suite
0,4945710333
0,4945710331
0,4945710331
Ahorn
Matlab
Test Suite
0,4952983876
0,4952983876
0,4952983876
a = x = 1000 Q (a, x) a = x = 10000 Q (a, x)
Ahorn
Matlab
Test Suite
0,4957947559
0,4957947558
0,4957947558
Ahorn
Matlab
Test Suite
0,4986701918
0,4986701917
0,4986701917
a = x = 100000 Q (a, x) a = x = 1000000 Q (a, x)
Ahorn
Matlab
Test Suite
0,4995794779
0,4995794779
0,4995794778
Ahorn
Matlab

Test Suite
0,4998670192
0,4998670196
0,4998670205

Tabelle E.1: Ausgewählte Eingabeparameter für die unvollständige Gamma-Funktion

x erfc (x) x erfc (x)
0,00-Test-Suite
Ahorn
Matlab
1,000000000000000
1,000000000000000
1,000000000000000
0,50-Test-Suite
Ahorn
Matlab
0,479500122186953
0,479500122186950
0,479500122186953
1,00-Test-Suite
Ahorn
Matlab
0,157299207050285
0,157299207050280
0,157299207050285
1,50-Test-Suite
Ahorn
Matlab
0,033894853524689
0,033894853524690
0,033894853524689
2,00 Test Suite 0,004677734981047 2,50 Test Suite 0,000406952017445
Maple 0,004677734981050 0,000406952017440 Maple
Matlab 0,004677734981047 0,000406952017445 Matlab
3,00-Test-Suite
Ahorn
Matlab
0,000022090496999
0,000022090497000
0,000022090496999
3,50-Test-Suite
Ahorn
Matlab
0,000000743098372
0,000000743098370
0,000000743098372

Tabelle E.2: Ausgewählte Eingangsparameter für die komplementäre Fehlerfunktion

So ist es evident, dass die verschiedenen Routinen für mathematische Ergebnisse, die ausreichend dicht beieinander liegen zu produzieren.
Die Unterschiede sind vernachlässigbar. Um die Wahrscheinlichkeit für den Erhalt einer ungenauen P-Value-Ergebnis, NIST reduzieren
vorgeschrieben hat empfohlen Eingabeparameter.

E-2

Analyse

Anhang F Unterstützende Software
Rank Berechnung von Binary Matrices

Anwenden elementarer Reihe, bei denen der Zusatz Betreiber ergriffenen Maßnahmen auf die Exklusiv-ODER-Verknüpfung ist.
Die Matrizen sind obere dreieckige Form mit nach vorne Zeilenoperationen reduziert, und der Vorgang wird
wiederholt in umgekehrter Reihenfolge in Verwendung rückwärts Zeilenoperationen, um zu einer Matrix in Dreiecksform kommen
zu bilden. Der Rang wird dann genommen, um die Anzahl der von Null verschiedenen Zeilen in der resultierenden Gaussian reduziert Matrix.

Vor Anwendung elementarer Row Operations:

Lassen Sie jedes Element in der m durch m-Matrix als ai, j bezeichnet werden

1. Set i = 1
2. Wenn das Element
(D. h. das Element auf der Diagonalen.
1), dann tauschen alle Elemente in der i-ten Zeile

mit allen Elementen in der nächsten Zeile, die eine enthält in der i
iiath Spalte (d. h. dieser Reihe ist der k-ten Zeile,
wo i <k =
m). Wenn keine Zeile enthält eine "1" in dieser Position, gehen Sie zu Schritt 4 fort.

3. Wenn das Element
iia
, Dann, wenn eine nachfolgende Zeile enthält eine "1" in der i-ten Spalte, ersetzen Sie jedes
Element in dieser Zeile mit der Exklusiv-ODER dieses Elements und das entsprechende Element in der

it

Reihe.

a. Set row = i + 1
b. Set col = i.
c. Wenn arow, col = 0, dann gehen Sie zu 3g Schritt.
d.
e. Wenn col = m, dann gehen Sie zu 3g Schritt.
f. col = col +1; gehen bis 3d Schritt.
g. Wenn row = m, dann gehen Sie zu Schritt 4 fort.
h. row = Zeile +1; gehen Sie zu Schritt 3b.
4. Wenn i <m-1, dann i = i +1; mit Schritt 2 fortfahren.
5. Vorwärts Zeilenoperationen abgeschlossen.
Rückwärts Anwendung elementarer Row Operations:
1. Set i = m.
2. Wenn das Element (d. h. das Element auf der Diagonalen.
1), dann tauschen alle Elemente in der i-ten Zeile

mit allen Elementen in der nächsten Zeile, die enthalten
IIAS eine in der i-ten Spalte (dh diese Zeile ist der k-ten Zeile,
wobei 1 =
k <i). Wenn keine Zeile enthält eine "1" in dieser Position, gehen Sie zu Schritt 4 fort.

3. Wenn das Element, dann, wenn alle vorherigen Zeile enthält eine "1" in der i-ten Spalte, ersetzen Sie jedes
Element in dieser Zeile mit der Exklusiv-ODER dieses Elements und das entsprechende Element in der

iia
it

Reihe.

F-1

d.
Ein statistischer Test SUITE für zufällige und Pseudo-Zufallszahlengeneratoren für kryptographische Anwendungen

a. Set row = i -1
b. Set col = i.
c. Wenn arow, col = 0, dann gehen Sie zu 3g Schritt.
e. Wenn col = 1, dann gehen Sie zu 3g Schritt.
f. col = col - 1; gehen bis 3d Schritt.
g. Wenn row = 1, dann gehen Sie zu Schritt 4 fort.
h. row = Zeile-1, geh zu Schritt 3b.

4. Wenn i> 2, dann i = i-1 und Schritt 2 fortfahren.
5. Rückwärts Reihe Vorgang abzuschließen.
6. Der Rang der Matrix = Anzahl der Nicht-Null-Zeilen.
Beispiel für Vorwärts Row Operations:

A
1 0 0 0 0 0
0 0 0 0 0 1
1 0 0 0 0 1
1 0 1 0 1 0
0 0 1 0 1 1
0 0 0 0 1 0
Die ursprüngliche Matrix.
B
1 0 0 0 0 0
0 0 0 0 0 1
0 0 0 0 0 1
0 0 1 0 1 0
0 0 1 0 1 1
0 0 0 0 1 0
Da a1, 1 = 1 und Zeilen 3 und 4 enthalten eine 1 in der ersten Spalte (siehe das Original matrix), werden die Zeilen 3 und 4 durch die Exklusiv-ODER-Verknüpfung, die Zeile und Zeile 1 ersetzt.
C
1 0 0 0 0 0
0 0 0 0 0 1
0 0 0 0 0 1
0 0 1 0 1 0
0 0 1 0 1 1
0 0 0 0 1 0
Da a2, 2.
1 und keine andere Zeile enthält eine "1" in dieser Spalte (siehe B), die
Matrix wird nicht verändert.

F-2

Analyse

1 0 0 0 0 0
0 0 0 0 0 1
0 0 1 0 1 0 Da a3, 3.
1, aber der 4. Zeile enthält eine "1" in der 3. Säule (siehe B oder C), die
D 0 0 0 0 0 1 zwei Reihen geschaltet sind.
0 0 1 0 1 1
0 0 0 0 1 0
1 0 0 0 0 0

0 0 0 0 0 1
0 0 1 0 1 0 Seit Zeile 5 enthält eine "1" in der 3. Säule (siehe D), Zeile 5 wird durch den
E 0 0 0 0 0 1 Exklusiv-ODER von Zeile 1 und Zeile 5 ein.
0 0 0 0 0 1
0 0 0 0 1 0
1 0 0 0 0 0
0 0 0 0 0 1
0 0 1 0 1 0 Da a4, 4.
1 und keine andere Zeile enthält eine "1" in dieser Spalte (siehe E), die
F 0 0 0 0 0 1-Matrix wird nicht verändert.
0 0 0 0 0 1
0 0 0 0 1 0
1 0 0 0 0 0
0 0 0 0 0 1
0 0 1 0 1 0 Da a5, 5.
1, aber Zeile 6 enthält eine 1 in Spalte 5 (siehe F), die beiden Reihen sind
G 0 0 0 0 0 1 geschaltet. Da keine Zeile unterhalb dieser enthält eine "1" in der 5.
Spalte, das Ende der
0 0 0 0 1 0 der Vorwärts-Prozess abgeschlossen ist.
0 0 0 0 0 1

Die Nachfolgende Backward Row Operations:

1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 1 0 Da a6, 6 = 1 und die Zeilen 2 und 4 enthalten diejenigen in der 6. Spalte
(siehe G), die Zeilen 2
H 0 0 0 0 0 0 und 4 sind durch die Exklusiv-ODER-Verknüpfung, die Zeile und Zeile 6
ersetzt.
0 0 0 0 1 0
0 0 0 0 0 1

F-3

Analyse

Ich
1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 1 0
0 0 0 0 0 1
Da a5, 5 = 1 und Zeile 3 enthält eine in der 5. Spalte (siehe H), Zeile 3 ist
ersetzt durch die Exklusiv-ODER-oder Zeile 3 und Zeile 5 ein.
J

1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 1 0
0 0 0 0 0 1
Da a4, 4.
1 und keine andere Reihe hat eine in Spalte 4, ist die Matrix nicht verändert.
K
1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 1 0
0 0 0 0 0 1
Da a3, 3 = 1, aber keine andere Reihe hat eine in Spalte 3, ist die Matrix nicht verändert werden.
L
1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 1 0
0 0 0 0 0 1
Da a2, 2.
1 und keine andere Reihe hat eine in Spalte 2 ist die Matrix nicht verändert,
und er abgeschlossen ist.

Da die endgültige Form der Matrix hat 4 Nicht-Null-Zeilen, ist der Rang der Matrix 4.

Bau Aperiodische Templates

Für die Zwecke der Durchführung der Non-überlappenden Template Matching statistischen Test, alle 2m m-Bit-Binär-
Sequenzen, die aperiodische sind, wurden im Voraus berechnet. Diese Vorlagen oder Mustern, wurden in eine Datei für m gespeichert
= 2 bis m = 21 ist. Der ANSI-C-Programm bei der Suche nach diesen Vorlagen verwendet wird weiter unten. Durch
Ändern Sie den Parameter M kann die Template-Bibliothek entsprechend der Vorlage konstruiert werden. This
Parameterwert nicht überschreiten B, da die DEC2BIN Konvertierungsroutine wird nicht korrekt funktionieren.
Denkbar ist, kann diesen Quellcode leicht geändert werden, um beliebige 2m m-Bit-Binär-Sequenzen für Bau
größere m.

Analyse

```
# Include
# Include <math.h>


# Define B 32
# Define M 6


int * A;
static long nichtperiodische;
unsigned displayBits (FILE *, long, long);


int main ()


{
FILE * fp1, * fp2;
lange i, j, count, num;


A = (unsigned *) calloc (B, sizeof (unsigned));
fp1 = fopen ("Template", "w");
fp2 = fopen ("dataInfo", "a");
num = pow (2, M);
count = log (num) / log (2);
nichtperiodische = 0;
for (i = 1; i <num; i + +)


displayBits (fp1, i, count);
fprintf (FP2, "M =% d \ n", M);
fprintf (FP2, "Anzahl der nichtperiodische Vorlagen =% u \ n", nichtperiodische);
fprintf (FP2, "# aller möglichen Vorlagen =% u \ n", num);
fprintf (FP2, "{# nichtperiodische} / {# Vorlagen} =% f \ n", (double) nichtperiodische /
num);
fprintf (fp2 ,"==========================================
========== \ n ");
fclose (fp1);
fclose (FP2);
frei (A);


return 0;
```

```
}


void displayBits (FILE * fp, long-Wert, Long Count)
{
int i, j, match, c, displayMask = 1 <<(B-1);


for (i = 0; i <B; i + +)
A [i] = 0;
for (c = 1; c <= B, c + +) {
if (value & displayMask)
A [c-1] = 1;
sonst.
A [c-1] = 0;
Wert <<= 1;
}


F-5



Analyse

for (i = 1; i <anzahl; i + +) {
match = 1;
if ((A [B-count]! = A [B-1]) & &


((A [B-count]! = A [B-2 ])||( A [B-count +1]! = A [B-1]))) {
for (c = B-count; c <= (B-1)-i, c + +) {


if (A [c]! = A [c + i]) {
match = 0;
break;


}


}
}
}
if (match) {


/ * Printf ("\ nPERIODIC TEMPLATE: SHIFT =% d \ n", i); * /
```

```
break;


}
}
if (! match) {


for (c = B-count; c <(B-1); c + +) fprintf (fp, "% u", A [c]);
fprintf (fp, "% u \ n", A [B-1]);
nichtperiodische + +;


}


return;
}
```

F-6

Analyse

Die Erzeugung der Binary Expansion der irrationalen Zahlen

Die Probe Mathematica-Programm in den Bau vier Beispiel-Dateien genutzt wird unten gezeigt.

Mathematica-Programm

```
(********************************************** **********)


(* Zweck: Wandelt num
seine Dezimaldarstellung mit *)
(* Seine binäre Darstellung. *)
(* *)
(* Achtung: Die $ MaxPrecision Variable muss auf * gesetzt werden)
(* Der Wert von d. Standardmäßig Mathematica *)

(* Setzt diese auf 50000, aber das kann erhöht werden .*)
(********************************************** **********)


BinExp [num_, d_]: = Module [{n, L},
```

```
Wenn [d> $ MaxPrecision, $ MaxPrecision = d];


n = N [num, d];


L = First [RealDigits [n, 2]]
];


SE = BinExp [E, 302500];
Save ["data.e", {SE}];


SP = BinExp [Pi, 302500];
Save ["data.pi", {SP}];


S2 = BinExp [Sqrt [2], 302500];
Save ["data.sqrt2", {S2}];


S3 = BinExp [Sqrt [3], 302500];
Save ["data.sqrt3", {S3}];
```

F-7

Analyse

Appendix G References

[1]
M. Abramowitz and I. Stegun, Handbook of Mathematical Functions, Applied Mathematics
Series. Vol. 55, Washington: National Bureau of Standards, 1964; reprinted 1968 by Dover
Publications, New York.
[2]
T. Cormen, C. Leiserson, & R. Rivest, Introduction to Algorithms. Cambridge, MA: The MIT
Press, 1990.
[3]
Gustafson et al., “A computer package for measuring strength of encryption algorithms,” Journal

of Computers & Security. Vol. 13, No. 8, 1994, pp. 687-697.
[4]
U. Maurer, “A Universal Statistical Test for Random Bit Generators,” Journal of Cryptology.
Vol. 5, No. 2, 1992, pp. 89-105.
[5]
A. Menezes, et al., Handbook of Applied Cryptography. CRC Press, Inc., 1997.
See http://www.cacr.math.uwaterloo.ca/hac/about/chap5.pdf.
[6]
W. Press, S. Teukolsky, W. Vetterling, Numerical Recipes in C : The Art of Scientific Computing, 2nd Edition. Cambridge University Press, January 1993.
[7]
G. Marsaglia, DIEHARD Statistical Tests: http://www.stat.fsu.edu/pub/diehard/.
[8]
T. Ritter, “Randomness Tests and Related Topics,
http://www.ciphersbyritter.com/RES/RANDTEST.HTM.
[9]
American National Standards Institute: Financial Institution Key Management (Wholesale),
American Bankers Association, ANSI X9.17 -1985 (Reaffirmed 1991).
[10]
FIPS 140-1, Security Requirements for Cryptographic Modules, Federal Information Processing
Standards Publication 140-1. U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, VA, 1994.
[11]
FIPS 180-1, Secure Hash Standard, Federal Information Processing Standards Publication 180-1.
U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, VA,
April 17, 1995.
[12]
FIPS 186, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186. U.S. Department of Commerce/NIST, National Technical Information Service,
Springfield, VA, May 19, 1994.
[13]
MAPLE, A Computer Algebra System (CAS). Available from Waterloo Maple Inc.;
http://www.maplesoft.com.