Alati za statističko testiranje nizova pseudoslučajnih brojeva

Petrović, Nebojša

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

ALATI ZA STATISTIČKO TESTIRANJE NIZOVA PSEUDOSLUČAJNIH BROJEVA

Diplomski rad

Nebojša Petrović

Osijek, 2018.

Sadržaj:

1. UVOD

Sami po sebi brojevi nisu slučajni. Tvrditi da je neki broj na primjer 120 slučajan baš i nema smisla. Spominjanjem slučajnih brojeva u biti se misli na nekoliko brojeva koji predstavljaju niz, te koji zadovoljavaju neke statističke uvjete slučajnosti. Nizovi slučajnih brojeva od velike su važnosti za praktičnu primjenu u kriptografiji. Uobičajeni kriptosustav koristi ključeve koji moraju biti proizvedeni na sasvim slučajan način, međutim generiranje uistinu slučajnog niza brojeva uopće nije jednostavan problem .

Prave slučajne brojeve vrlo je teško dobiti jer se u svakodnevnom životu pojavljuju u prirodnim pojavama i situacijama koje je apsolutno ne moguće predvidjeti. Zbog sve veće potrebe za slučajnim brojevima potrebno ih je dobiti na neki drugi način i to uz pomoć računala odnosno generatora. Generirani brojevi dobiveni pomoću generatora slučajnih brojeva su statistički nezavisni i nepredvidivi. Međutim, pošto se takvi brojevi generiraju uz pomoć neke vrste algoritma ipak se može zaključiti da ti brojevi nisu sasvim slučajni, zbog toga ih je pravilnije nazvati pseudoslučajnim brojevima. Pseudoslučajni brojevi generirani pomoću računalnog algoritma koji se na prvi pogled mogu se zaista činiti slučajnima, ali to zapravo nisu.

U ovom diplomskom radu opisani su neki od poznatijih algoritama za generiranje pseudoslučajnih brojeva koji se mogu koristiti u mnoge svrhe, uključujući kriptografiju, modeliranje i simulaciju aplikacija. Osim toga opisat ćemo osnovne pojmove bitne za kriptografiju i kriptoanalizu te standardizirane statističke testove za ispitivanje slučajnosti generiranih pseudoslučajnih nizova, a kao praktični rad testirati zadane nizove brojeva i utvrditi dali zadovoljavaju standardne testove slučajnosti i dali se uopće mogu nazvati pseudoslučajni brojevi.

2. SLUČAJNI I PSEUDOSLUČAJNI BROJEVI

Kada se govori o slučajnim brojevima, vrlo bitna stavka je da ta da je niz slučajnih brojeva apsolutno nemoguće predvidjeti, međutim ako se slučajni brojevi dobivaju pomoću računala koji je deterministički uređaj, te u kojemu se ne bi trebalo događati ništa što se ne može predvidjeti možemo zaključiti da brojevi koji su nastali na takav način ipak nisu u potpunosti slučajni, iako se na prvi pogled zaista tako čine. Iz tog razloga slučajan brojevni niz koji je nastao upotrebom računala odnosno nekog algoritma točnije je nazvati „pseudoslučajnim“ nizom brojeva.

(Pseudo)Slučajni brojevi u današnje vrijeme vrlo često se koriste i primjenjuju se u raznim područjima kao što su:

1) Simulacije

Prilikom računalne simulacije prirodnih procesa slučajni brojevi su od velike važnosti kako bi sama simulacija bila što više realnija. Simulacije se koriste u raznoraznim područjima od nuklearne fizike (raspadanje radioaktivne tvari odvija se na slučajan način) do operacijskog istraživanja (vremenski interval do dolaska sljedećeg kupca u trgovinu je nepredvidljiv).

2) Biranje test uzoraka

Ispitivanje svih mogućih slučajeva koji se mogu pojaviti vrlo često nije praktično iz tog razloga slučajni odabir manjeg broja uzoraka daje tipični uzorak koji predstavlja cjelinu.

3) Numerička analiza

Rješavanje nekih teških numeričkih problema poprilično je lagano i precizno uz pomoć slučajnih brojeva.

4) Programiranje

Neki sasvim slučajno odabrani podatci, u svrhe programiranja, vrlo često su dobar izvor podataka koji se koriste prilikom testiranja programa.

5) Donošenje odluka

U svakodnevnim situacijama kada je potrebno izabrati samo jednu odluku od nekoliko jednako vrijednih odluka u većini slučajeva poseže se za slučajnim izborom.

6) Kriptografija

Korištenje slučajnih brojeva u kriptografiji od velike je važnosti. Kako bi se ostvarila sigurna komunikacija preko nesigurnog komunikacijskog kanala potrebno je koristiti slučajne brojeve pomoću kojih se generira ključ za šifriranje poruka.

7) Zabava

Slučajnost je prisutna i u svijetu igara, od kockanja, do miješanja karata i računalnih igara.

8) Pravosudni sustav

U pravosudnom sustavu u SAD-u tijekom sudskih procesa važnu ulogu imaju porotnici koji se slučajno biraju među građanima koji imaju pravo glasa i koji svojim odlukama utječu na presudu.

Vratimo se malo u natrag, slučajni brojevi koristili su se od 1927. godine kao velike tabele slučajnih brojeva. Nešto kasnije za potrebe lutrije u Velikoj Britaniji osmišljen je poznati stroj naziva *„ERNIE“*. S pojavom računala pojavila se ideja da se slučajni brojevi generiraju uz njihovu pomoć. John von Neumann 1946. godine predložio je prvi generator slučajnih brojeva koji koristi metodu *sredine kvadrata (eng. Middle-square method)*. Metoda je prilično trivijalna, uzima se *n*-teroznamenkasti prirodni broj, koji se kvadrira te nakon toga se uzima srednjih *n* znamenaka dekadskog zapisa tog rezultata. Postupak se ponavlja s tim da bi uvijek uzimao broj srednjih *n* znamenaka, a u slučaju ako bi znamenaka bilo manje, dodaje se potreban broj nula s lijeve strane zapisa. Korištenjem ove metode može se dobiti jako dobar niz pseudoslučajnih brojeva, međutim postoji jedan problem. Ukoliko je sredina jednaka nuli svaki sljedeći član niza je jednak nuli, što nikako ne odgovara poimanju slučajnog brojevnog niza.

Na primjer kvadrat broja 5772156649 je 33317792380594909201, a njegova sredina ako je n=10 daje sljedeći „slučajni broj“ : 7923805949. Kvadriranjem dobivenog broja dobivamo sljedeći „slučajni broj“. Korištenjem ove metode može se dobiti jako dobar niz pseudoslučajnih brojeva, međutim postoji jedan problem. Ukoliko je sredina jednaka nuli svaki sljedeći član niza je jednak nuli, što nikako ne odgovara poimanju slučajnog brojevnog niza.

Na primjer, početni broj je 62 a n=2. Dobivamo sljedeće:

- $62^2=3844$

- $84^2 = 7056$
- $5^2=0025$
- $2^2=0004$
- $0^2=0000$
- $0^2=0000$ i tako dalje.

Dobiveni niz „slučajnih“ brojeva pokazuje da je potrebna izvjesna pažnja prilikom odabira prirodnog broja. [1]

2.1. Pokazatelji slučajnosti

1) Matematičko očekivanje

Matematički očekivana vrijednost odnosno očekivanje slučajne varijable X u teoriji vjerojatnosti interpretira se kao srednja (očekivana) vrijednost od X, koja se definira kao broj $\mathbb{E}$ [X]:

$$\mathbb{E}[X] = \sum_{x \in ImX} x\, f_X(x) \quad \text{(ako je X diskretna funkcija)} \tag{2-1}$$

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x f_X(x) dx \quad \text{(ako je X ne prekidna funkcija)} \tag{2-2}$$

Ukoliko je g : $\mathbb{R} \rightarrow \mathbb{R}$ neka realna funkcija (koja se na nekom dijelu funkcije ne prekida) i ako je X : $\Omega \rightarrow \mathbb{R}$ slučajna varijabla, tada je g(X) = g ◦ X : $\Omega \rightarrow$ R isto slučajna varijabla. Tada možemo izračunati $\mathbb{E}$ [g(X)] pomoću slijedećih jednadžbi [2]:

$$\mathbb{E}[g(X)] = \sum_{x \in ImX} g(x)\, f_X(x) \quad \text{(ako je X diskretna funkcija)} \tag{2-3}$$

$$\mathbb{E}[g(X)] = \int_{-\infty}^{\infty} g(x) f_X(x) dx \quad \text{(ako je X ne prekidna funkcija)} \tag{2-4}$$

Uz pomoć navedenih formula moguće je pokazati da matematičko očekivanje slučajne varijable X ima svojstvo linearnosti. Za realne funkcije g_1, g_2,..., g_k i brojeve c_1, c_2,..., c_k vrijedi:

$$\mathbb{E}\left[\sum_{i=1}^{k} c_i\ g_i\ (X)\right] = \sum_{i=1}^{k} c_i\ \mathbb{E}[g_i(X)] \tag{2-5}$$

2) Varijanca i standardna devijacija

Varijanca slučajne varijable X predstavlja mjeru raspršenja vrijednosti od matematičkog očekivanja, odnosno srednje kvadratno odstupanje varijable X od $\mathbb{E}[X]$.

$$\mathbb{V}ar[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] \tag{2-6}$$

Zbog toga što matematičko očekivanje ima svojstvo linearnosti vrijedi slijedeća jednadžba:

$$\mathbb{V}ar[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2] \tag{2-7}$$

Standardna devijacija od X računa se kao:

$$\sigma(X) = \sqrt{\mathbb{V}ar[X]} \tag{2-8}$$

Fizikalna jedinica kojom se prikazuje raspršenje standardne devijacije ista je onoj pomoću koje se prikazuju vrijednosti varijable X.

Idealni generator pseudoslučajnih brojeva treba generirati brojeve koji simuliraju neprekidnu uniformnu razdiobu gdje slučajna varijabla X u granicama ⟨α, β⟩, ima slijedeću gustoću razdiobe:

$$f_X(x) = \begin{cases} \frac{1}{\beta-\alpha} & za\ x \in <\alpha, \beta>, \\ 0 & inače. \end{cases} \tag{2-9}$$

U tom slučaju se matematičko očekivanje računa prema formuli:

$$\mathbb{E}[X] = \frac{\alpha+\beta}{2} \tag{2-10}$$

A varijanca po formuli:

$$\mathbb{V}ar[X] = \frac{(\beta-\alpha)^2}{2} \tag{2-11}$$

Svojstvo neprekidne uniformne razdiobe je to da pod intervali funkcije jednake duljine imaju jednake vjerojatnosti.

3) Entropija

Entropija predstavlja mjeru nepredvidljivosti diskretne slučajne varijable odabranu iz nekog skupa mogućih vrijednosti. Unutar tog skupa već je definirana vrijednost odabira slučajne varijable. Vrlo često se koristi Shannonova entropija koja izražava, u ovom slučaju, realizaciju

slučajne varijable odnosno niz pseudoslučajnih brojeva. Neka je *X* diskretna slučajna varijabla na konačnom skupu *S* sa distribucijom vjerojatnosti p. Entropija H(X) slučajne varijable X definira se kao:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) \text{ [bit/simbolu]} \quad (2\text{-}12)$$

Ukoliko su izlazne varijable generatora u potpunosti slučajne njihova entropija mora biti jednaka ukupnoj duljini poruke zbog toga što je vjerojatnost svakog od simbola jednaka. Entropija se izražava u bitovima ukoliko koristimo algoritam s bazom 2, međutim moguće je odabrati bili koju bazu. Definicija entropije vrlo je korisna jer odgovara onome što intuitivno shvaćamo kao sadržaj informacije.

2.2 . Pravi slučajni brojevi

Kao sto smo već napomenuli, brojevi sami po sebi nisu slučajni. Govoreći o slučajnim brojevima zapravo se misli na niz brojeva koji zadovoljava neke uvjete slučajnosti, na primjer nepredvidljivost i uniformnost.

- Za niz brojeva reći ćemo da je nepredvidljiv ukoliko znajući bilo koji početak tog niza nikako ne možemo predvidjeti sljedeći član tog istog niza (osim naravno potpuno slučajno). Stoga, ukoliko znamo X_0, X_1,X_2… X_n , ne možemo ni na koji način predvidjeti sljedeći član niza a to je X_{n+1} bez obzira na to koliko članova niza n je poznato i koliko god velik n bio.

- Za niz brojeva možemo reći da je uniforman ako se u tom nizu sve vrijednosti pojavljuju u podjednakim omjerima odnosno čija je vjerojatnost pojavljivanja ista. Na primjer, ukoliko promatramo dovoljnu dugačak niz brojeva dobivenih bacanjem standardne igraće kockice sa brojevima od 1 do 6, očekujemo da se u tim nizu pojave svi brojevi od 1 do 6 te da se niti jedan jedini broj u nizu ne pojavljuje puno puta više ili manje od nekog drugog broja. Bacimo li tu istu kockicu tisuću puta i evidentiramo sve brojeve koji su se pojavili na gornjoj strani, nikako ne možemo analizirajući zapisane brojeve saznati koji broj će se pojaviti prilikom sljedećeg bacanja. [1]

Osim primjera bacanja igraće kockice možemo navesti još jedan, a to je izvlačenje brojeva u igri na sreću odnosno *„Loto“*. Dulji vremenski period možemo pratiti izvučene brojeve ali bez obzira na prethodna kola nećemo biti u mogućnosti predvidjeti brojeve koji će se izvući u sljedećem kolu, jedino ih možemo sasvim slučajno pogoditi. Stoga vođenje bilo kakve statistike i evidencije prethodnih kola bilo kojeg Loto izvlačenja baš i nema smisla. Bacanje igraće kockice i izvlačenje lota dva su primjera izvora pravih slučajnih nizova brojeva. Ujedno su i primjeri gdje svaki broj ima istu vjerojatnost da se pojavi ili bude izvučen. Tako da, sljedeći puta kada budete uplaćivali loto ukoliko se neki broj pojavio u zadnjih 5, 10 ili 15 kola ne mora značiti da će se pojaviti ponovno ili se pak nije pojavio u prethodnih 5,10 ili 15 kola ne mora značiti da će se pojaviti u idućem.

Naime, u Italiji se dogodilo da se broj 53 u Loto izvlačenju nije pojavio skoro pune dvije godine. Razmišljajući na način da se mora pojaviti talijani su potrošili oko 3.5 bilijuna eura ulagajući novac na broj 53 koji se iz mjeseca u mjesec uporno nije pojavljivao. Ovisnici o broju 53 odnosni oni uporni koji su svako kolo uplaćivali novac na broj 53 upali su u dugove, izgubili kuće, obitelji i ušteđeni novac, osim toga potvrđeno je i nekoliko smrtnih slučajeva kasnije prozvanih žrtvama *„Venice 53“*. Nedugo nakon toga broj je bio izvučen a država je morala isplatit oko 600 milijuna eura dobitnicima pogođenog broja. Iz ovog primjera možemo zaključiti da je apsolutno nemoguće predvidjeti sljedeći broj niza koji je pravi slučajni niz bez obzira na poznate prethodne brojeve niza.

2.3. Pseudoslučajni brojevi

Promatrajući svijet oko sebe možemo primijetiti da se razni događaji na prvi pogled čine nepovezani i slučajni, razlog tome može biti nedostatak informacija o tim događajima ili ne mogućnosti da shvatimo povezanost istih. Slučajni odnosno pseudoslučajni brojevi najčešće su dobivaju uz pomoć računala ili nekih postupaka koje je u mislima ne moguće rekonstruirati. Iz tog razloga brojevi koji nisu istinski slučajni nama se ipak kao da jesu.

Na primjer, napisan je slijedeći niz slučajnih brojeva:

21, 28, 21, 19, 13, 14, 9, 24, 16, 27, 4, 1, 14, 19, 13, 19, 13, 29.

Ovakav niz brojeva na prvi pogled čini se potpuno slučajan i brojevi se ne čine povezanim, međutim ovaj niz nije slučaj. Naime, svaki broj u navedenom nizu predstavlja poziciju slova u hrvatskoj abecedi, u navedenom nizu zapravo piše *„ovonijeslučajniniz“*.

Zbog toga što su istinski slučajni brojevi vrlo rijetki, a u svijetu se javlja sve veća i veća potreba za slučajnim brojevima koji se koriste u raznorazne svrhe potrebno ih je dobiti na neki drugi način. Taj problem je naveo matematičare da korištenjem računala dobiju brojeve koji će na prvi pogled izgledati zaista slučajno. Međutim dobiveni brojevi pomoću računala u većini slučajeva temelje se na rekurzivnim aritmetičkim ili logičkim formulama. Svaki generator koji za računanje sljedećeg elementa niza koristi već prije izračunatu vrijednost niza (jednu ili više) zahtjeva početnu vrijednost tog istog niza. Ovisno o kvaliteti algoritma postupak dobivanja slučajnog niza na takav način moguće je rekonstruirati, ali bez poznavanja parametara, konstanti i matematičkih funkcija koje su se koristile prilikom generiranja broja to je poprilično zahtjevan i težak zadatak. Svaki generator slučajnih brojeva potrebno je ispitati, a svojstva koja se ispituju su: period ponavljanja niza, uniformnost raspodjele brojeva te raspodjela n-torki dobivenih od slučajnog niza n. Ukoliko postoji generator koji je već korišten i testiran te koji zadovoljava navedena svojstva, preporučuje se primijeniti isti u svrhu generiranja slučajnih brojeva. Zbog toga što se novi generator koji je tek programiran mora podvrgnuti testiranjima kojih ima jako puno te za koje je potrebno jako dugo vremena.

3. POSTUPCI I ALGORITMI ZA GENERIRANJE PSEUDOSLUČAJNIH BROJEVA

Generator brojeva je nekakav algoritam ili uređaj čiji je izlaz niz brojeva koji su statistički potpuno nezavisni i ne predvidivi. Generatori se općenito dijele na dvije skupine:

1) Generatori istinski slučajnih brojeva
2) Generatori pseudoslučajnih brojeva

Generatori istinski slučajnih brojeva temelje se na fizičkim postupcima pomoću kojih se promatraju i bilježe nepredvidljive i slučajne pojave u prirodi. Međutim, dizajnirati sklopovski uređaj i/ili računalni program pomoću kojega će se prikupljaju vrijednosti neke pojave imajući na umu da ista ne smije biti predvidljiva vrlo je težak zadatak. Ovakva vrsta generatora većinom je sklopovske naravi a izvedba je osjetljiva i skupa, dok samo generiranje niza brojeva može biti vrlo sporo. Generatori istinski slučajnih brojeva mogu se podijeliti u dvije skupine:

1) Sklopovski izvori :
   - Termički šum poluvodičke diode ili električkog otpornika
   - Frekvencijska nestabilnost oscilatora
   - Naboj kondenzatora nakon zadanog perioda punjenja
   - Zvuk iz mikrofona ili slika iz kamere
   - Vrijeme radioaktivnog raspada čestica

Sklopovski generatori za generiranje slučajnih brojeva u većini slučajeva koriste neke nepredvidive fizikalne pojave, međutim te fizikalne pojave i alati koji se koriste za mjerenje doprinose tome da izlaz iz takvog generatora nije uniformno raspoređen zbog asimetričnih i sistemskih odstupanja koja se pojavljuju. Kod sklopovskih izvora vrlo često je potrebno dobivene podatke programski obraditi jer su generatori periferni uređaji koji se fizički moraju povezati sa računalom.

2) Programski izvori:
   - Vrijeme između dva pritiska tipki na tipkovnici računala ili miša
   - Sistemski sat

- Sadržaj ulaznog ili izlaznog spremnika
- Specifične varijable operacijskog sustava

Generatori pseudoslučajnih brojeva zbog determinističke prirode računala generiraju determinističke algoritme. U ovisnosti o unosu slučajne varijable veličine k bitova, kao izlaz generator vraća niz brojeva veličine I>>k. Generiran niz na takav način doista se čini potpuno slučajnim. Početna varijabla koja se unosi kao ulaz u generator u većini slučajeva naziva se sjeme, a izlaz se naziva pseudoslučajni niz brojeva. Kod ovakve vrste generatora bitno je primijetiti da se za jednake unose odnosno vrijednosti sjemena na izlazu pojavljuju isti nizovi koji se ponavljaju nakon određenog vremenskog perioda.

Kao što smo već spomenuli prvi generator pseudoslučajnih brojeve predložio je John von Neumann 1946. godine koji koristi metodu *sredine kvadrata*. Ova metoda temelji se na jednostavnosti i na tome da je slučajan niz brojeva moguće dobiti bez korištenja računala, međutim generirani niz u većini slučajeva nije kvalitetan. U današnjem svijetu velika većina programskih jezika u sebi imaju ugrađene funkcije za generiranje slučajnih brojeva, ali također kvaliteta izlaznog niza nije toliko dobra odnosno statistička svojstva niza su poprilično loša, dok funkcije unutar operacijskih sustava daju puno kvalitetnije slučajne brojevne nizove. Svaki pseudoslučajni generator koji kao izlaz ima pseudoslučajan niz mora zadovoljiti određene uvjete i imati određena svojstva kako bi se uopće mogao nazvati pseudoslučajni niz brojeva.

Svojstva koja mora zadovoljavati niz koji je generiran pseudoslučajnim generatorom su slijedeća:

- Brojevi u prostoru moraju biti ravnomjerno raspoređeni, a vjerojatnost pojavljivanja svakog broja mora biti približno jednaka. Ovo svojstvo treba vrijediti i za sve generirane pod nizove koji su generirani pomoću originalnog niza.

- Kod nizova dobivenih pomoću generatora pseudoslučajnih brojeva ne smije postojati korelacije, odnosno niti jedan pod niz ne smije ovisiti o nekom drugom pod nizu gdje je pod niz, niz dobiven pomoću originalnog niza.

- Period pseudoslučajnog generatora treba biti jako velik. Niz brojeva generira se pomoću računala to jest generatora koji prolazi kroz puno različitih stanja. Premda postoji jako

puno različitih stanja u jednom trenutku stanja će se početi ponavljati jer je broj stanja ipak konačan. Period pseudoslučajnih generatora u ovom slučaju je od 2^{31} do 2^{64}.

Glavna razlika između generatora pseudoslučajnih brojeva i generatora istinski slučajnih brojeva je ta da se kod izlaza iz pseudoslučajnog generatora brojevi nakon određenog vremenskog perioda počinju periodično ponavljati, dok se generator istinski slučajnih brojeva ponaša na drugačiji način odnosno svi brojevi su slučajni i potpuno nepredvidivi.

3.1. Linear Congruential Generator – LCG

Linearni kongruencijski generator *(eng. Linear Congruential Generator – LCG)* 1948. godine osmislio je *Derrick Henry Lehmer*. LCG još se može nazvati i linearni slijedni generator i jedan je od najpoznatijih generatora pseudoslučajnih brojeva. Aritmetička formula pomoću koje se dobiva niz cijelih brojeva je:[4]

$$X_{n+1} = (aX_n + c) mod\ m\ , n = 0,1,2, \dots \quad (3\text{-}1)$$

Gdje je :

X_n- n-ti slučajni cijeli broj

X_{n+1} - n+1 slučajni cijeli broj

X_0 - sjeme, početna vrijednost

a - koeficijent (multiplikator)

c - konstanta (inkrement)

m - gornja granica brojeva (modul za modulsku aritmetiku)

Sljedeći slučajni broj X_{n+1} generira se korištenjem prethodnog slučajnog broja X_n koji se množi sa konstantom a, nakon toga se na taj broj dodaje konstanta c. Kada je broj $(aX_n + c)$ generiran primjenjuje se modulo arimetika uz parametar m kako bi se dobio broj X_{n+1} zatim se postupak ponavlja. Međutim za sam početak odnosno prvi broj ovakvog niza ova metoda zahtjeva početnu vrijednost X_0, to jest sjeme.

Početno sjeme se može dobiti na sljedeće načine:

1) Uz pomoć nekog istinski slučajnog generatora kao što je sistemski sat
2) Može biti postavljeno kao konstanta, zadana vrijednost ili pak neka unutrašnja vrijednost
3) Određen od strane korisnika

Unaprijed poznate konstante a,c,m i početno sjeme X_0 parametri su pomoću kojih se određuje generirani niz brojeva. Zbog toga se LCG može prikazati kao uređena četvorka: *LCG* (m,a,c,X_0).

Aritmetika koja se koristi za generiranje ovakvog niza brojeva a je vrlo jednostavna, te ju je pomoću programa lako izvesti. Međutim, zbog jednostavne aritmetike kojom se generiraju slučajni brojevi potrebno je pripaziti pri korištenju ovakvog načina generiranja, jer će se za isto početno sjeme X_0 uvijek dobiti identičan niz brojeva kao i kada se prvi puta generira niz što je dobra pogodnost za testiranje programa. Pošto je svaki broj određen svojim cjelobrojnim prethodnikom zahtjevi ovog generatora za memorijom su minimalni što LCG čini brzim. Iako je ovaj generator brz, što je korisno u različitim simulacijama, on nije dovoljno siguran. Radi jednostavnosti kojom je ovaj generator izveden period je ograničen, te se na može dogoditi da period niza bude veći od konstante m.

Pomoću cijelih brojeva dobivenim pomoću LCG-a vrlo lako možemo dobiti realne slučajne brojeve i to na način:

$$R_n = \frac{X_n}{m} \quad ili \quad R_n = \frac{X_n}{(m-1)} \tag{3-2}$$

Maksimalan period LCG-a je m, ali uz pomoć različitog kombiniranja parametara moguće je postići period koji je manji od m. Za varijablu m u većini slučajeva se uzimaju potencije broja dva ili prosti brojevi, dok se za sjeme X_0 najčešće koristi veličina dobivena pomoću sistemskog sata računala. Vrijednost varijable a treba biti ne paran broj, a jako veliki brojeve potrebno je uzimati bez predznaka. Najjednostavniji kriterij kvalitete ovakvih generatora je taj da prosječna vrijednost svih brojeva treba težiti ka polovici gornje granice intervala .

Ovisno o parametrima, linearni kongruencijski generator općenito je moguće podijeliti u tri skupine:[5]

1) $m = 2^M, c > 0$

   Konstanta M je potencija broja 2, dok je konstanta c pozitivan cijeli broj. Ovakva vrsta generatora je karakterizirana velikom brzinom rada. Kako bi se dobio puni period 2^M potrebno je izabrati konstantu *a* tako da je $a = 1 \ (mod \ 4)$ i konstanta *c* mora biti neparna . Kod ovakve vrste generatora manje značajni bitovi nisu u potpunosti slučajni pa se može uočiti određena pravilnost.

2) $m = 2^M, c = 0$

   Konstanta M ista je kao i kod prethodnog generatora međutim konstanta c je u ovom slučaju jednaka nuli. Pošto je konstanta *c* jednaka nuli, zbog toga se ovakav generator naziva MCG-u *(eng. Multiplicative congruential generator).* Najveći mogući period je 2^{M-2} , ukoliko za a vrijedi jednadžba a = 3 (mod 8) ili a = 5 (mod 8). Također kao i kod prethodnog generatora, niti u ovom slučaju manje značajni bitovi nisu potpuno slučajni.

3) $m = p \ (prost \ broj)$

   U ovom slučaju maksimalan period je $p - 1$ i to ako je konstanta *a* također prost broj. U ovakvoj vrsti generatora, bitovi sa manjim značajem mogu ali i ne moraju biti slučajni.

Najpoznatiji LCG generator, stvoren je od strane IBM-a 1968.godine, naziva se RANDU. Parametri generatora su $RANDU(2^{31}, 65539,0,1)$ i odabrani su na način da generator bude dovoljno brz i jednostavan za izvedbu.

Još neki od poznatih LCG generatora su:

- *„Minimalni standard“ LCG* $(2^{31-1}, 16807, 0, 1)$
- *Super-Duper LCG* $(2^{32}, 69069, 0, 1)$
- *SIMSCRIPT LCG* $(2^{31-1},630360016, 0,1)$
- *APPLE LCG* $(2^{35}, 1220703125, 0, 1)$
- *DRAND48 LCG* $(2^{48}, 25214903917, 11, 0)$
- *CRAY LCG* $(2^{48}, 44485709377909, 0, 1)$

3.2. Lagged Fibonacci generator – LFG

LFG je vrsta generatora koja se vrlo često koristi a ime je dobio zbog toga što je jednadžba vrlo slična rekurzivnoj jednadžbi pomoću koje se generiraju brojevi Fibonacci-jevog niza.

Opći oblik formule za generiranje slučajnih brojeva na ovakav način je[5]:

$$X_n = X_{n-p} \circ X_{n-q} \tag{3-3}$$

Gdje su p i q cijeli brojevi, p mora biti veći od q i q mora biti veći od nule, a ° je binarna operacija (zbrajanje, množenje, mod m, XOR, …). U većini slučajeva upotrebljava se modulo *m* aritmetika, gdje se *m* potencija broja dva, tako da će formula imati sljedeći oblik:

$$X_n = (X_{n-p} + X_{n-q}) \, mod \, 2^M \tag{3-4}$$

LFG generator može se prikazati kao uređena trojka: *LFG (p,q,M)* .

Početni uvjet generatora je niz brojeva $X_0,\ldots\ldots X_{n-1}$ koje je na primjer moguće dobiti pomoću LCG generatora. Ukoliko se parametri p i q pažljivo odaberu postići će se duži periodi u odnosu na periode LCG generatora. Period P računa se kao:

$$P = (2^p - 1) * 2^{M-1} \tag{3-5}$$

Dva najčešće upotrjebljivana LFG generatora su:

1. LFG(17,5,31) gdje je period $P \approx 2^{47}$
2. LFG(55,24,31) gdje je period $P \approx 2^{85}$

Određenim ispitivanjima uspješno je dokazano da nizovi brojeva generiranih uz pomoć LFG-a imaju bolja slučajna svojstva u odnosu na one generirane uz pomoć LCG-a. Također je kod LFG-a primijećen pravilnost grupiranja grupiranju *n*-torki unutar u *n* dimenzionalnog prostora. Osim toga postoji problem zahtijevanja većeg memorijskog prostora od strane LFG generatora prilikom generiranja brojeva.

Linearni kongruencijski generator i Lagged Fibonacci generator su dvije osnovne vrste generatora pseudoslučajnih brojeva koji se unatoč tome da niz generiranih brojeva prikazuje

jako puno nepravilnosti u današnje vrijeme vrlo često koriste u svrhu računalne simulacije gdje je brzina vrlo važan faktor. LCG i LFG primjenjuju se iz razloga što je njihova izvedba temelji na jednostavnosti, minimalnim zahtjevima za memoriju, dok je dobiveni niz brojeva dovoljno slučajan.

Za korištenje slučajnih brojeva u kriptografiji brzina nije ključan faktor, nego sigurnost. Generirani slučajni brojevi koji se koriste za potrebe šifriranja moraju biti ne predvidivi, a ulazna vrijednost generatora treba biti takva da se ne može otkriti jednostavnim metodama pogađanja. Veliki broj različitih nizova slučajnih brojeva postiže se velikim periodom za koji je poželjno da je što veći. Generatori slučajnih brojeva koji su kriptografski sigurni, odnosno koji se mogu koristiti za šifriranje bili bi također korisni za korištenje u računalnim simulacija, međutim algoritmi za generiranje takvih nizove su složeni pa samim time je i izvedba znatno sporiju u odnosu na standardne generatore. U slijedećim poglavljima biti će opisana dva generatora kojima je ključan faktor sigurnost, zbog toga se i koriste u kriptografije svrhe,a to su ANSI X9.17 generator i RSA generator.

3.3. ANSI X9.17 generator

ANSI X9.17 generator slučajne brojeve generira pomoću DES algoritma. Ovaj algoritam koristi trostruko *E-D-E* šifriranje s dva DES ključa, a šifriranje se definira na sljedeći način[6]:

$$E(x) = E_{K3}\big(D_{K2}(E_{K1}(x))\big) \tag{3-6}$$

gdje $E_K(x)$ predstavlja šifriranje pomoću ključa *K*, a $D_K(x)$ predstavlja dešifriranje pomoću ključa *K*. Algoritam predstavlja trostruku E-D-E šifriranje ukoliko vrijedi :

$$K_1 = K_3 \tag{3-7}$$

Ulazni podaci ANSI X9.17 generatora su slučajno i tajno 64-bitno sjeme, cijeli broj *m* i DES ključevi za kriptiranje *K*. Rezultat ovog generatora to jest izlaz je 64-bitni niz pseudoslučajnih brojeva $x_1, x_2, x_3, \ldots, x_m$.

Generiranje slučajnog niza se vrši se prateći slijedeće korake:

1) Izračunati srednju vrijednost I

$$I = E_K(D) \tag{3-8}$$

gdje je D predstavlja 64-bitni zapis vremena

2) za *i = 1 do m* potrebno je izračunati:

$$x_i = E(I \oplus s) \tag{3-9}$$

$$s = E(x_i \oplus I) \tag{3-10}$$

Ovaj generator koristi se u mnoge svrhe iako nije dokazano da je sto posto siguran za kriptografske svrhe. Za razliku od kriptografski sigurnih generatora pseudoslučajnih brojeva ovaj generator odlikuje se većom brzinom i jednostavnijom izvedbom.

3.4. RSA generator

RSA generator je kriptografski siguran generator pseudoslučajnih brojeva, a njegova se sigurnost temelji na problemu faktorizacije broja n. Iz skupa $\mathbb{Z}_n$ odabire se sjeme te se zatim formira niz elemenata iz skupa $\mathbb{Z}_n$ gdje je svaki novi element niza RSA enkripcija prethodnog elementa niza. Najmanje značajni bit svakog elementa u nizu čini generirani niz slučajnih bitova.

Algoritam RSA generatora se odvija prema slijedećim koracima[7]:

1) Neka su *p* i *q* prosti brojevi s (*k/2*) – bitova
2) Potrebno je izračunati umnožak $n=p \cdot q$
3) Neka je *b* prirodan broj relativno prost s $\rho(n)$
4) *n* i *b* su javni, *p* i *q* su tajni
5) Sjeme s_0 je element od $\mathbb{Z}_n$ (s_0 ima *k* bitova). Za $i \geq 0$ računa se

$$s_{i+1} = s_i^b \ mod \tag{3-11}$$

$$f(s_0) = (z_1, z_2, \dots, z_l) \tag{3-12}$$

$$z_i = s_i \ mod \ 2 \tag{3-13}$$

Rezultat RSA generatora je niz od z_1 do z_n koji predstavljaju slučajne bitove. Generiranje slučajnih brojeva pomoću RSA algoritma znatno je sporije u odnosu na LCG, LFG i ANSI X9.17 generatore, zbog potrebe računanja potencije velikog broja te ostatak dijeljenja sa velikim brojem. Međutim određenim metodama moguće je ubrzati rad ovog generatora uz minimalno narušavanje kvalitete generiranog niza. Prilikom korištenja metoda za ubrzavanje generiranja brojeva pozornost treba obratiti na oprez pri optimizaciji, jer pojava korelacije između susjednih bitova uveliko povećava mogućnost neželjenog upada.

4. KRIPTOGRAFIJA

Kriptografija je znanost "tajnog pisanja", odnosno znanost pohrane informacija u formi koja je čitljiva samo onome kome je informacija namijenjena dok će za ostale biti neupotrebljiva. Generatori slučajnih brojeva imaju važnu ulogu u kriptografiji. Velika većina kriptografskih algoritma i protokola traže u nekim fazama rada nizove slučajnih brojeva, kao što su generiranje slučajnih bitova pri postupku autentifikacije, potom kao ključ u digitalnom potpisu te najviše pri generiranju javnih i tajnih ključeva, gdje je cilj da budu neprobojni radi sigurnosti podataka koji se šalju.

Za razliku od steganografije gdje se tajnost poruke dobiva na način da se ista sakrije unutar teksta ili slike, u kriptografiji tajnost poruke se dobiva njenim modificiranjem. Sama riječ kriptografija nastala je iz riječi kripta što znači tajna i grafija što znači pisanje, riječi su grčkog podrijetla a doslovan prijevod bi bio tajnopis. U 5.stoljeću prije Krista kod starih Grka pojavili su se neki elementi kriptografije. Spartanci su izmislili drveni štap za šifriranje nazvan skital koji se koristio za tajnu komunikaciju. Na njega bi se namotala vrpca na koji se okomito napisala poruka, nakon toga vrpcu je potrebno odmotati te bi na istoj ostala izmiješana poruka, a koju može pročitati samo onaj tko posjeduje štap čija je debljina jednaka onom štapu pomoću kojega je poruka napisana. Osim spartanaca kriptografiju je koristio jedan od prvih vojskovođa, to jest Gaj Julije Cezar. On je poruke slao tako što je sva ili pojedina slova pomjerao za tri ili više mjesta u abecedi, takvu poruku mogao je pročitati samo onaj tko je poznavao „pomjeri za“ pravilo. Šifrirana Cezarova izjava prilikom prelaska Rubikona glasila bi: fqkf ofkhz kyz, ali ukoliko pomaknemo svako slovo za šest mjesta lako se može pročitati Alea iacta est, odnosno kocka je bačena. Jedna od najpoznatijih naprava za šifriranje je ENIGMA koju je 1918. godine izumio Artur Scherbius. Pošto je ENIGMA imala jako puno mogućih kombinacija sa šifriranje (čak 150 738 274 937 250) mnogi su mislila da je nemoguće pronaći način za dekriptiranje poruke šifrirane pomoću ENIGME, ipak Marian Rejewski sa poljskom grupom kripotoanalitičara i Alan Turing, sa britanskom grupom pronašli su način za dekriptiranje. Razbijanje šifre kombinacijom kriptoanalize i klasične špijunaže imalo je vrlo važnu ulogu za tijek i ishod drugog svjetskog rata.

4.1. Osnovni pojmovi kriptografije

Temeljni zadatak kriptografije jest omogućiti dvjema osobama komuniciranje preko nesigurnog komunikacijskog kanala poput telefonske linije ili računalne mreže na način da treća osoba koja možda nadzire komunikacijski kanal nije u mogućnosti razumjeti poruke koje se šalju tim komunikacijskim kanalom . U kriptografskoj literaturi osoba koja šalje poruku naziva se Alice, osoba koja prima poruku Bob, a njihov protivnik odnosno napadač Oskar ili Eva.

Osnovni pojmovi koji se vežu uz kriptografiju su sljedeći:

- Otvoreni tekst *(eng. Plaintex)*
- Ključ *(eng. Key*)
- Kriptografski algoritam *(eng.cipher)*
- Šifriranje *(eng.encryption)*
- Šifrat *(eng. Ciphertext)*
- Dešifriranje *(eng. Decryption)*

Podatci koji je potrebno kriptirati nazivaju se otvoreni tekst a sadržaj teksta je bilo kakva vrsta podatka koju pošiljalac želi poslati kao na primjer nekakav tekst, slika, numerički podatci ili bilo što drugo. Pošiljalac pomoću unaprijed dogovorenog ključa transformira otvoreni tekst koristeći određene matematičke funkcije koje se nazivaju kriptografski algoritmi. U ovom dijelu procesa slanje poruke preko nesigurnog komunikacijskog kanala generatori slučajnih brojeva imaju jako veliku ulogu jer se ključevi generiraju baš pomoću takvih generatora. Postupak pomoću kojeg se otvoreni tekst transformira koristeći ključeve i algoritme naziva se šifriranje a dobiveni rezultat šifrat ili kriptogram. Pošiljalac nakon šifriranja preko komunikacijskog kanala primaocu šalje samo šifrat. Ukoliko postoji treća osoba koja nadzire komunikacijski kanal u mogućnosti je doznati samo šifrat, ali bez ključa ne može doznati otvoreni tekst. Za razliku od napadača, primalac koji posjeduje ključ pomoću kojega je šifrirana poruku može dešifrirati šifrat i odrediti otvoreni tekst.

Slika 4.1. Proces prijenosa poruke pomoću kriptografije

Kriptografija kao znanstvena disciplina čija je temeljna svrha osigurati pouzdan i siguran prijenos šifriranih podataka mora osigurati sljedeće[8]:

1) Integritet *(eng. Data integrity)*
   Integritet zahtjeva da samo ovlašteni korisnici smiju mijenjati informacije, te ujedno mora postojati nekakav način na koji će se provjeriti dali je neovlaštena osoba promijenila informaciju ili nije.

2) Tajnost *(eng. Confidentiality)*
   Tajnost informacija koje se šifriraju i šalju smiju biti dostupne samo određenim ovlaštenim osobama

3) Autentifikacija *(eng. Authentification)*
   Autentifikacija se vrši na dvije razine: razina korisnika i razina informacije. Prije početka rada a početak rada potrebno je utvrditi identitet korisnika da bi se vidjelo ima li korisnik pravo pristupa te se određuje i sigurnosna razina rada. Na razini informacije, autentifikacija označava provjeru odakle dolazi ta informacija, tko je pošiljatelj, kada je stigla, tip informacije te ostale značajke.

4) Odgovornost *(eng. Responsibility)*
   Odgovornost čini jako veliku ulogu u današnjem svijetu zbog toga što se veliki dio novčanih sredstava obavlja putem interneta.

Kriptografski algoritam je matematička funkcija koja se koristi prilikom šifriranja i dešifriranja. Funkcije preslikavaju osnovne elemente otvorenog teksta u osnovne elemente šifrata

i obratno. U ovisnosti o ključu funkcije se biraju iz određene familije funkcija. Skup svih mogućih ključeva naziva se prostor ključeva, a vrijednosti ključeva, prostor ključeva. Svaki kriptosustav sastoji se od kriptografskog algoritma, otvorenih tekstova, šifrata i ključeva. Definicija kriptosustava je sljedeća[9]:

Kriptosustav je uređena petorka $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ za koju vrijedi:

- $\mathcal{P}$ je konačan skup svim mogućih jasnih tekstova
- $\mathcal{C}$ je konačan skup svih mogućih šifrata
- $\mathcal{K}$ je konačan skup svih mogućih ključeva
- Za svaki $K \in \mathcal{K}$ postoji algoritam šifriranja $e_K \in \mathcal{E}$ i odgovarajući algoritam dešifriranja $d_K \in \mathcal{D}$. Pritom su funkcije $e_K : \mathcal{P} \rightarrow \mathcal{C}$ i $d_K : \mathcal{C} \rightarrow \mathcal{P}$ funkcije sa svojstvom da je $d_K(e_K(x)) = x$ za svaki otvoreni tekst $x \in \mathcal{P}$.

Kriptosustavi se klasificiraju s obzirom na sljedeća tri kriterija:

1) Vrsta operacija koja se koristi prilikom šifriranja

   Kriptosustavi koji ovise o vrsti operacije koji se prilikom šifriranja koristi dijele se na supstitucijske i transpozicijske šifre. Kod supstitucijskih šifri element jasnog teksta zamjenjuje se sa nekim drugim elementom, dok se kod transpozicijskih šifri elementi jasnog teksta premještaju. Također postoje i kriptosustavi koji kombiniraju ove dvije metode šifriranja.

2) Način obrade otvorenog teksta

   Kriptosustavi ovisni o načinu na koji se otvoreni tekst obrađuje dijele se na blokovne i protočne šifre. Kod blokovnih šifri se obrađuje jedan po jedan blok elemenata otvorenog teksta koristeći jedan te isti ključ za razliku od protočnih gdje se elementi otvorenog teksta obrađuju jedan po jedan ali pri tome koriste niz ključeva koji se paralelno generira.

3) Tajnost i javnost ključeva

   Kriptosustavi dijele se na simetrične i asimetrične kriptosustave. Kod simetričnih kriptosustava, ključ za dešifriranje može se izračunati poznavajući ključ za šifriranje i obratno. U većini slučajeva ti ključevi su jednaki. Sigurnost simetričnih kriptosustava leži u tajnosti ključa zbog toga se još nazivaju i *kriptosustavi s tajnim ključem*. Kod asimetričnih kriptosustava ključ za dešifriranje se ne može (barem ne u nekom razumnom vremenu) izračunati iz ključa za šifriranje. ključ za šifriranje je u ovom slučaju *javni ključ*. Pomoću javnog ključa bilo tko može šifrirati poruku, ali samo osoba koja ima odgovarajući ključ za dešifriranje (*privatni* ili *tajni ključ*) može dešifrirati tu poruku.

4.2. Kriptoanaliza

Kriptoanaliza je grana znanosti koja se razvijala paralelno sa kriptografijom. Glavni cilj kriptoanalize je odgonetnuti stvaran sadržaj šifrirane poruke bez poznavanja ključa. Pokušaj kriptonaliziranja neke šifrirane poruke ili bilo koje druge informacije smatra se kao napad. Jedna od osnovnih pretpostavki je da kriptoanalitičar zna koji se kriptosustav koristi prilikom šifriranja. Ova pretpostavka ne mora uvijek biti točna , ali nije baš pogodno da sigurnost sustava leži na pretpostavci da napadač ne zna tu informaciju.

Postoje četiri osnovne razine kriptoanalitičkih napada na šifrirane poruke. [9]

1) Samo šifrat

   Kriptoanalitičar posjeduje šifrat od nekoliko poruka koje su šifrirane istim algoritmom. Na temelju toga treba otkriti otvoreni tekst ili ključ pomoću kojega su se poruke šifrirale.

2) Poznat jasan tekst

   Kriptoanalitičar posjeduje šifrat neke poruke njemu odgovarajući jasan tekst. Na temelju toga treba otkriti ključ.

3) Odabrani otvoreni tekst

   Kriptoanalitičar je dobio privremeni pristup alatu koji je korišten prilikom šifriranja, te može uzeti neki tekst, šifrirati ga i na taj način i saznati šifrat.

4) Odabrani šifrat

   Kriptoanalitičar ima pristup alatu koji je korišten prilikom dešifriranja, može uzeti šifrat i doći do jasnog teksta. Ovakva vrsta napada tipična je kod kriptosustava koji koriste javni ključ. Zadatak kriptoanalitičara je saznati tajni ključ.

5) *Potkupljivanje, ucjena, krađa i slično*

   Ovakva vrsta napad ne spada doslovno u kriptoanalizu, ali se pokazala vrlo efikasna i često primjenjivan vrsta napada u kombinaciji s "pravim" kriptoanalitičkim napadima.

Algoritam koji se koristi u kriptografiji je siguran ukoliko ne daje dovoljno informacija bitnih za kriptoanalizu šifriranog teksta, bez obzira na količinu šifriranog teksta. Kriptografski jak algoritam je onaj koji se ne može probiti uporabom sadašnjih ili budućih metoda u realnom vremenu.

5. STANDARDIZIRANI TESTOVI SLUČAJNOSTI

Standardizirani testovi slučajnosti služe za ispitivanje kvalitete generatora pseudoslučajnih brojeva to jest slučajno generiranih nizova brojeva. Testovi slučajnosti uvelike pomažu prilikom detektiranja nekih slabosti koje generator pseudoslučajnih brojeva možda ima. Otkrivanje određenih pogrešaka postiže se na način da se generirani niz uzet kao uzorak izlaza generatora podvrgava različitim statističkim testovima pomoću kojih je moguće ustanoviti dali generirani niz brojeva pokazuje neke od poželjnih karakteristika niza ili ne. Svrha svakog od standardiziranih testova je izvući samo one bitne karakteristike niza (kojih ima beskonačno) pomoću kojih se može zaključiti dali se niz ponaša kao „slučajan“ ili ne. Rezultat svakog pojedinog testa ne može se uzeti kao konačan rezultat nego kao vjerojatnost. Iz tog razloga testirani niz potrebno je podvrgnuti svim ostalim statističkim testovima, ukoliko generator ne zadovoljava određene testirane karakteristike niza isti može biti odbačen kao ne slučaj, nasuprot tome ukoliko zadovoljava odnosno prolazi sve testirane karakteristike može se sa velikom vjerojatnošću tvrditi da je generator slučajan.

5.1. ENT skup statističkih testova

Autor skupa statističkih testova naziva ENT objavio je John Walker 2008. godine. ENT je jedan od skupa statističkih testova koji se vrlo često koriste, a sadrži različite testove slučajnosti pomoću kojih se testiraju nizovi slučajnih brojeva formirani u bajtove koji su pohranjeni u nekoj datoteci. Ovi testovi izrazito su korisni i moćni te se koriste za procjenjivanje slučajnosti pseudoslučajnih nizova generiranih pomoću generatora pseudoslučajnih brojeva, također se koriste za testiranje slučajnosti različitih algoritmima za kompresiju i drugih aplikacije gdje je gustoća informacija i slučajnost u području interesa.

ENT skup statističkih testova sastoji se od ukupno 6 testova a to su : [10]

1) Entropija *(eng. Entropy)*
   Rezultat testa entropije je gustoća informacija sadržaja neke datoteke koja je izražena kao broj bitova po znaku.

2) Hi-kvadrat test *(eng. Chi-square test)*

   Hi-kvadrat test je najčešće korišten test koji se koristi za provjeru slučajnosti podataka, a izuzetno je osjetljiv na pogreške nizova generiranih uz pomoć generatora slučajnih brojeva. Hi-kvadrat distribucija izračunava se za niz bitova u testiranom nizu podataka te se rezultat izražava se kao apsolutni broj odnosno postotak koji pokazuje koliko često istinski slučajan niz brojeva premašuje izračunatu vrijednost. Postotak se tumači kao broj do kojega se testirana sekvenca sumnja da nije slučajna. Ukoliko je postotak veći od 99 ili manji od 1, slijed je gotovo sigurno nije slučajan. Ukoliko je postotak između 99 i 95 ili između 1 i 5, slijed je sumnjiv. Postotci između 90 i 95 i 5 i 1 ukazuju da je slijed "gotovo sumnjiv".

3) Aritmetička sredina *(eng. Arithmetic Mean)*

   Ovaj test predstavlja jednostavno zbrajanje svih bajtova/bitova u generiranom nizu koji se dijele sa ukupnom duljinom niza. Rezultat je srednja vrijednost srednja vrijednost svih bitova u nizu koji bi trebao biti blizu 0.5.

4) Monte Carlo vrijednost za Pi *(eng. Monte Carlo Value for Pi)*

   U ovom testu uzima se sekvenca od 6 bajtova koja predstavlja 24-bitne X i Y koordinate koje se nalaze unutar kvadrata. Ukoliko je udaljenost od slučajno generirane točke manja od radijusa kruga upisanog u kvadrat, tada se sekvenca od 6 bajtova smatra „pogotkom“. Postotak „pogodaka“ može se iskoristiti za računanje vrijednosti Pi . Ukoliko je vrijednost bliža pravoj vrijednosti broja Pi niz se može smatrati slučajnim.

5) Serijski koeficijent korelacije *(eng. Serial Correlation Coefficient)*

   Ovaj test otkriva u kojoj mjeri svaki bajt u generiranom nizu ovisi o prethodnom bajtu. Za slučajne nizove, ova vrijednost (koja može biti pozitivna ili negativna) će ,naravno, biti jako blizu nule.

5.2. DIEHARD(ER) skup statističkih testova

DIEHARD(ER) je skup statističkih testova koje je razvio američki profesor George Marsaglia. Testovi su razvijani nekoliko godina a 1995.godine objavljeni su prvi puta kao cjelina. DIEHARD skup testova tada se sastojao od 16 vrlo jakih statističkih testova koji služe za

provjeru slučajnosti nizova brojeva. Ovi testovi kao imaju rezultat imaju, takozvane p-vrijednosti čija je vrijednost od nula do jedan. DIEHARD set sastoji se od slijedećih testova: Razmak između rođendana, ponavljajuće permutacije, pozicije matrica 31x31 i 32x32, pozicije matrica 6x8, test ponavljanja na 20-bitnoj riječi, test ponavljanja OPSO i OQSO, DNA test, broj jedinica u nizu bitova, broj jedinica u segmentiranom nizu, test parkirališta, test minimalne udaljenosti, test slučajnih sfera, test stiskanja, test preklapajućih suma, test smjera i test igre s dvije kocke. Ovi testovi su vrlo bitni za ispitivanje slučajnosti generiranih nizova i pokazali su se vrlo moćnima u otkrivanju ne slučajnosti i potvrđivanju slučajnosti. Jako puno softverskih i hardverskih generatora za koje se tvrdilo da su dobri i pouzdani pali su nekim od navedenih testova. Na primjer linearni kongruencijski generator i lagged Fibonacci generator vrlo često su padali na ispitima te su se vodili kao ne slučajni generatori.

Trenutno se u DIEHARDER setu nalazi ukupno 6 statističkih testova i to su [11] :

1) Test smjera *(eng. Runs test)*

   Test smjera ponavlja se kroz 10 000 vrijednosti ukupno deset puta i utvrđuje uzastopne smjerove, gdje su elementi niza u porastu vrijednosti ili u smanjenju vrijednost, rezultat se nalazi između 0 i 1 i zatim se uspoređuje sa poznatim matricom kovarijancije.

2) Razmak između rođendana *(eng. Birthdays test)*

   U ovom testu odabire se m rođendana u godini od n dana, uz navođenje međuprostora između rođendana. Nakon toga se određuje najbolja odgovarajuća linija te se određuje odstupanje od očekivane funkcije distribucije. Odstupanjem se mjeri kvaliteta testa.

3) Test minimalne udaljenosti *(eng. Minumum distance test)*

   Ovaj test provodi se 100 puta odabire se n=8000 slučajnih točaka u kvadratu te se traži d, odnosno udaljenost između svih parova točaka. Udaljenost bi trebali biti eksponencijalno distribuirani. Vrijednost p daje odstupanje od očekivane vrijednosti.

4) Test 3D sfere *(eng. 3D Spheres test)*

   U ovom testu odabire se 4000 slučajnih točki koji su smješteni u kocku veličine 1000*1000. U svakoj točki se nalazi centar sfere dovoljno velike samo da dotakne najbližu točku. Test se ponavlja 20 puta i pronalazi najmanji radijus. Radijusi bi trebali

biti eksponencijalno distribuirani, a vrijednost p se određuje na temelju toga koliko su dobiveni rezultati različiti od očekivanih.

5) Statistički test jedinica *(eng. STS Monobit test)*
   Ovaj test broji jedinične bitove u cijelom nizu slučajnih brojeva. Rezultat se uspoređuje sa očekivanim brojem p. Ovaj test je vrlo djelotvoran u otkrivanju otvorenih i slabih generatora.

6) Statistički test smjera *(eng. STS Runs test)*
   Ovaj test broji ukupan broj ponavljanja nula i jedinica po uzorku bita. Korak nula mora početi sa 10 i završiti sa 01 dok korak jedan mora početi sa 01 i završiti sa 10. Test se kako bi se utvrdilo dali generirani niz ima binomnu razdiobu gdje je p =0.25.

5.3. NIST skup statističkih testova

Nacionalni institut za standard i tehnologiju *(eng. National Institute of Standards and Technology)* razvio je statistički skup testova koji se sastoji od ukupno 15 testova razvijenih za testiranje slučajnosti bitova ili brojeva dobivenih pomoću generatora slučajnih brojeva. Testovi su konstruirani na način da provjeravaju različite vrste ne slučajnosti koje se u nekog generiranom nizu slučajnih brojeva mogu pojaviti. Većina testova ima standardnu ili hi-kvadrat distribuciju kao referentnu. U slučaju da testirani niz nije slučajan, statistika testa će pasti u ekstremnim područjima referentne distribucije. Standardna normalna distribucija se koristi za usporedbu vrijednosti statistike testa dobivene uz pomoć generatora slučajnosti s očekivanom vrijednošću statistike pod pretpostavkom slučajnosti.

NIST-ov set statističkih testova sastoji se od[12] :

1) Frekvencijski test *(eng. Frequency monobit test)*

Svrha testiranja:

Cilj frekvencijskog testa je uočiti dali je broj 1 i 0 testiranog niza približno jednak. Pošto svi drugi testovi ovise o prolazu ovoga testa, možemo reći da je pozitivan rezultat frekvencijskog testa nužan, ali ne dovoljan za donošenje odluke dali je testirani niz zaista slučajan ili ne.

Poziv funkcije:

Frequency(n), gdje je :

n - duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

S_{obs} – apsolutna vrijednost zbroja X_i (gdje je $X_i = 2\varepsilon - 1 = \pm 1$) u sekvenci koji se dijeli sa kvadratnim korijenom ukupne duljine niza.

Referentna distribucija je „pola“ normalna (za veliki n). Ako je $z = \frac{S_{obs}}{\sqrt{2}}$ distribuirana kao normalna razdioba tada je |z| distribuirana kao polu normalna razdioba. Ako je ulazna sekvenca slučajna, tada će pozitivne i negativne jedinice imati tendenciju da se međusobno ponište, tako da statistika teste bude vrlo blizu nule. je slučajnih, a plus i minus one će imati tendenciju da se otkazuju jedna drugu, tako da statistika testa bude oko 0. Ako postoji previše jedinica ili previše nula, tada će statistička ispitivanja biti veća od nule.

Opis testa:

(1) Pretvorba u ±1 : jedinice i nule ulazne sekvence (ε) pretvaraju se u vrijednosti -1 i +1, te se zatim zbrajaju $S_n = X_1 + X_2 + \cdots + X_n$, gdje je $X_i = 2\varepsilon - 1 = \pm 1$

Na primjer, ako je $\varepsilon = 1011010101$, tada je $n = 10$ i $S_n = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + (-1) + 1 = 2$

(2) Računanje statistike testa:

$$S_{obs} = \frac{|S_n|}{\sqrt{n}} \tag{5-1}$$

$$S_{obs} = \frac{|2|}{\sqrt{10}} = 0.632455532.$$

(5-2)

(3) Računanje P-vrijednosti $P = erfc\left(\frac{S_n}{\sqrt{n}}\right)$, gdje je ercf komplementarna error (pogreška) funkcija

$$P = erfc\left(\frac{0.632455532}{\sqrt{10}}\right) = 0.527089. \quad (5\text{-}3)$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je dobivena P vrijednost ≥ 0.01 , možemo zaključiti da je sekvenca slučajna. Ukoliko je vrijednost P mala (<0.01), možemo zaključiti da je $|S_n|$ ili $|S_{obs}|$velik. Velike pozitivne vrijednosti S_n indiciraju previše jedinica, dok negativne vrijednosti S_n indiciraju previše nula.

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 100 bitova.

2) Frekvencijski test unutar bloka *(eng. Frequency test within a block)*

Svrha testiranja:

Cilj frekvencijskog testa unutar bloka je provjeriti odnos 0 i 1 unutar M-bitnih blokova. Potrebno je uočiti dali je u svakom M-bitnom bloku broj jedinica i nula približno jednak., odnosno dali je učestalost jedinica u M-bitnim blokovima približno jednaka M/2 . Za blok čija je veličina 1 ovaj test se pretvara u frekvencijski test.

Poziv funkcije:

BlockFrequency (M,n), gdje je:

M - duljina pojedinog bloka

n - duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

χ^2(obs) – mjera koliko dobro promatrani udio jedinica unutar danog M-bitnog odgovara očekivanom omjeru koji iznosi (1/2).

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Ulazna sekvenca dijeli se u blokove $N = \left\lfloor \frac{n}{M} \right\rfloor$ koji se ne preklapau. Neiskorišteni bitovi na kraju sekvence se odbacuju.

Na primjer, ako je n=10, M=3 i ε = 0110011010, kreirat će se ukupno 3 bloka (N=3), koji se sastoje od 011, 001 i 101, posljednja 0 se odbacuje

(2) Određuje se udio π_i jedinica u svakom M-bitnom bloku koristeći jednadžbu

$$\pi_i = \frac{\sum_{j=1}^{M} \varepsilon_{(i-1)M+j}}{M} \tag{5-4}$$

Za prethodnu sekvencu $\pi_1 = 2/3$, $\pi_2 = 1/3$, i $\pi_3 = 2/3$.

(3) Računa se $\chi 2$ distribucija:

$$\chi 2(\text{obs}) = 4M \sum_{i=1}^{N} \left(\pi_i - \frac{1}{2}\right)^2 \tag{5-5}$$

$$\chi 2(\text{obs}) = 4 \cdot 3 \cdot \left(\left(\frac{2}{3} - \frac{1}{2} \right)^2 + \left(\frac{1}{3} - \frac{1}{2} \right)^2 + \left(\frac{2}{3} - \frac{1}{2} \right)^2 \right) = 1$$

(5-6)

(4) Potrebno je izračunati P- vrijednost

$$\mathrm{P} = \text{igamc} \left(\frac{\mathrm{N}}{2} \right), \frac{\chi^2(\text{obs})}{2}$$

(5-7)

gdje je igamc nekompletna gama funkcija

$$P = igamc \left(\frac{3}{2}, \frac{1}{2} \right) = 0.801252.$$

(5-8)

Pravilo odlučivanja (na razini od 1%) :

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je dobivena P vrijednost ≥ 0.01 , možemo zaključiti da je sekvenca slučajna. Male vrijednost P (<0.01) indiciraju velike devijacije od podjednakog broje jedinica i nula u minimalno jednom bloku.

Preporučena duljina unosa:

Preporuka je da se svaka sekvenca koja se ispituje sastoji od najmanje 100 bita. Bitno je primijetiti da je n≥MN. Veličinu bloka M treba odabrati tako da je M ≥20, M> 0.01n i N <100.

3) Test izvođenja *(eng. Runs test)*

Svrha testiranja:

Promatrana karakteristika ovog testa je ukupan broj jedinica i nula koji se pojavljuju u nizu, gdje je niz ne prekinuti slijed istih bitova. Niz je duljine *k*, što znači da se niz sastoji od točno *k* istih

bitova, a omeđen je prije i poslije sa bitom suprotne vrijednosti. Ovim testom moguće je utvrditi dali su oscilacije između takvih pod nizova prebrze ili pak prespore.

Poziv funkcije:

Runs(n), gdje je:

n – duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \dots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

$V_n(obs)$ - ukupan broj izvođenja (zbroj izvođenja nula + zbroj izvođenja jedinica)

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

Test izvođenja provodi frekvencijski test kao preduvjet.

(1) Izračunati proporciju л jedinica u ulaznoj sekvenci : $\pi = \frac{\sum_j \varepsilon_j}{n}$

Na primjer, ako je $\varepsilon = 1001101011$, tada je n=10, $\pi = \frac{3}{5}$

(2) Provjeriti dali sekvenca prolazi frekvencijski test, ako je $|\pi - \frac{1}{2}| \geq \tau$, tada se test ne mora provesti (jer ne prolazi frekvencijski test). Ukoliko test nije primjenjiv vrijednost P postavlja se na 0.0000. Varijabla $\tau = \frac{2}{\sqrt{n}}$, je predefinirana u testnom kodu.

Budući da je $\tau = \frac{2}{\sqrt{10}} = 0.63246$, te $\left|\pi - \frac{1}{2}\right| = \left|\frac{3}{5} - \frac{1}{2}\right| = 0.1 < \tau$, pošto je promatrana vrijednost л unutar granica, test izvođenja se može primijeniti.

(3) Izračunati statistiku testa $V_n(\text{obs}) = \sum_{k=1}^{n-1} r(k) + 1$, gdje je r(k)=0 ako je $\varepsilon_k = \varepsilon_{k+1}$, inače r(k) =1

$$V_{10}(\text{obs}) = (1+0+1+0+1+1+1+1+0)+1 = 7 \qquad (5\text{-}9)$$

(4) Izračunati vrijednost P

$$P = erfc\left(\frac{|V_n(obs) - 2n\pi(1-\pi)|}{2\sqrt{2n}\,\pi(1-\pi)}\right) \qquad (5\text{-}10)$$

$$P = erfc\left(\frac{7 - \left(2\cdot 10\cdot\frac{3}{5}\cdot\left(1-\frac{3}{5}\right)\right)}{2\cdot\sqrt{2\cdot 10}\cdot\frac{3}{5}\cdot\left(1-\frac{3}{5}\right)}\right) = 0.147232. \qquad (5\text{-}11)$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je dobivena P vrijednost ≥ 0.01 možemo zaključiti da je sekvenca slučajna. Velike vrijednosti V_n(obs) indicirale bi da je oscilacija u sekvenci prebrza, dok bi mala vrijednost indicirala da je oscilacija prespora (oscilacija se smatra kao promjena jedinice na nulu i obratno). Brze oscilacije predstavljaju velik broj promjena, dok spore oscilacije predstavljaju malo broj promjena. Na primjer sekvenca 010101010 oscilira za svaki bit dok sekvenca koja sadrži 100 jedinica,zatim 73 nule, te 127 jedinica (ukupno 300 bitova) ima samo 3 izvođenja (3 promjene), dok se očekuje barem 150.

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 100 bitova.

4) Test za najduži niz jedinica u bloku *(eng. Test for the longest run of ones in a block)*

Svrha testiranja:

Karakteristika koja se promatra u ovom testu je najdulji niz bita 1 u bloku u M-bitnom bloku. Svrha ovog testa je odrediti da li je duljina najduljeg niza jedinica unutar ispitivane sekvence u skladu s dužinom najdužeg trajanja koji se može očekivati od generatora slučajnih brojeva. Bitno je napomenuti da moguće nepravilnosti kod očekivane duljine najduljeg niza jedinica upućuju na to da kod najduljeg niza nula također postoji nepravilnost. Međutim dugi nizovi nula se ne provjeravaju posebno.

Poziv funkcije:

LongestRunsOfOnes(n), gdje je:

n – duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

M – duljina svakog bloka, kod test je predefiniran tako da podržava ukupno tri vrijednosti za M a to su M=8, M=128 i $M=10^4$ koji se odabiru u ovisnosti o duljini testirane sekvence,n:

Tablica 5.3.1. Definirani vrijednosti M u odnosu na odabrani n

Minimalan n	**M**
128	8
6272	128
750000	10^4

N- broj blokova, odabran u ovisnosti o odabranoj duljini bloka M

Ispitivanje statistike testa i referentna distribucija:

χ^2(obs) – mjera koliko dobro promatran niz ponavljanja bitova unutar danog M-bitnog bloka odgovara očekivanom najduljem nizu ponavljanja unutar M-bitnih blokova.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Podijeliti sekvencu u M-bitne blokove

(2) Frekvencije v_i najduljih nizova jedinica u svakom bloku podijelit će se u kategorije gdje svaka ćelija sadrži broj ponavljajućih nizova jedinica određene duljine

Za vrijednosti M koje su podržane od strane testnog koda, v_i ćelije će sadržavati sljedeće vrijednosti:

Tablica 5.3.2. Prikaz vrijednosti v_i ovisno o odabranom M

v_i	*M=8*	*M=128*	*M=10^4*
v_0	*≤1*	*≤4*	*≤10*
v_1	*2*	*5*	*11*
v_2	*3*	*6*	*12*
v_3	*≥4*	*7*	*13*
v_4		*8*	*14*
v_5		*≥9*	*15*
v_6			*≥16*

(3) Potrebno je izračunati $\chi^2(\text{obs}) = \sum_{i=0}^{k} \frac{(v_i - N\pi_i)^2}{N\pi_i}$, gdje su vrijednosti K i N određene uz pomoć varijable M prema slijedećoj tablici

Tablica 5.3.3. Vrijednosti M ovisne o vrijednostima K i N

M	*K*	*N*
8	*3*	*16*
128	*5*	*49*
10^4	*6*	*75*

(4) Potrebno je izračunati vrijednost P

$$P = igamc\left(\frac{K}{2}, \frac{\chi^2(obs)}{2}\right) \tag{5-12}$$

Primjer:

K=3, M=8, $\varepsilon =$ 11001100000101010110110001001100111000000000001001
00110101010001000100111101011010000000110101111100
11001110011011011000010110010

n=128

Tablica 5.3.4. Prikaz ponavljanja uzastopnih jedinca unutar bloka

Broj ponavljanja uzastopnih jedinica unutar bloka	**Broj ponavljanja uzastopnih jedinica unutar bloka**
11001100 (2)	00010101 (1)
01101100 (2)	01001100 (2)
11100000 (3)	00000010 (1)
01001101 (2)	01010001 (1)
00010011 (2)	11010110 (2)
10000000 (1)	11010111 (3)
11001100 (2)	11100110 (3)
11011000 (2)	10110010 (2)

v_0= 4, v_1 =4, v_2 =3, v_3 = 3, v_4 =0, $\chi^2 = 4.882457$ Dobivena vrijednost P=0.180609

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je vrijednost P ≥ 0.01 (P=0.180609) zaključaj je da je sekvenca slučajna. Velike vrijednosti χ^2 (obs) ukazuju na to da testirana sekvenca sadrži velike grupe jedinica, odnosno da se sastoji on nekoliko blokova u kojoj se jedinice pojavljuju jedna iza druge.

Preporučena duljina unosa:

Preporuka je da svaka testirana sekvenca sadrži onoliko bitova koliko je definirano u tablici 5.3.1.

5) Test binarnog ranga matrice *(eng. Binary matrix rank test)*

Svrha testiranja:

Karakteristika koja se promatra u ovom testu je rang matrice dobivene iz pod niza testiranog originalnog niza. Svrha testa je provjera linearne ovisnosti između fiksne duljine pod nizova originalnog niza.

Poziv funkcije:

Rank(n), gdje je

n – duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

M - broj redova u svakoj matrici (predefinirano, iznosi 32)

Q – broj kolona u svakoj matrici (predefinirano, iznosi 32)

Ispitivanje statistike testa i referentne distribucije:

$\chi^2(obs)$ - mjera koliko dobro promatran broj ranga matrice u testiranoj sekvenci odgovara očekivanom broju redova pod pretpostavkom slučajnosti.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Ulazna sekvenca sekvencijalno se dijeli u razdvojive M*Q blokove, gdje je $N = \left[\frac{n}{MQ}\right]$ odbačeni bitovi se ne koriste, dok se sakupljeni M*Q bitni segmenti prebacuju u matricu MQ gdje je svaki red matrice popunjen sa Q-bitnim blokovima iz izvorne ε sekvence.

Na primjer, ako je n=20, M=Q=3 i ε= 01011001001010101101 tada je $N = \left[\frac{n}{3*3}\right] = 2$, što bi značilo da postoje 2 matrice po 3 reda sa po 3 stupca (M*Q=3*3=9) . Zadnja dva bita (0 i 1) se odbacuju.

Dvije dobivene matrice su $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ i $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

(2) Odrediti binarni rank (R_l) svake matrice gdje je l=1,…,N.

Za prethodni primjer rank prve matrice je $R_1 = 2$, a druge je $R_2 = 3$.

(3) Neka je F_M = broj matrica čiji je rank $R_l = M$

F_{M-1} = broj matrica čiji je rank $R_l = M-1$

$N-F_M - F_{M-1}$ = broj preostalih matrica

Za prethodni primjer $F_M = F_3 = 1$ (rank $R_2 = 3$) , $F_{M-1} = F_2 = 1$ (rank $R_1 = 2$), te niti jedna ostala matrica nema manji rank.

(4) Potrebno je izračunati $\chi^2(\text{obs})$

$$\chi^2(\text{obs}) = \frac{(F_M - 0.2888N)^2}{0.2888N} + \frac{(F_{M-1} - 0.5776N)^2}{0.5776N} + \frac{(N - F_M - F_{M-1} - 0.1336N)^2}{0.1336N} \quad (5\text{-}13)$$

$$\chi^2(\text{obs}) = \frac{(1 - 0.2888\cdot 2)^2}{0.2888\cdot 2} + \frac{(1 - 0.5776\cdot 2)^2}{0.5776\cdot 2} + \frac{(2-1-1 - 0.1336\cdot 2)^2}{0.1336\cdot 2} = 0.596953 \quad (5\text{-}14)$$

(5) Potrebno je izračunati vrijednost P

$$P = e^{\frac{-\chi^2(\text{obs})}{2}} \tag{5-15}$$

$$P = e^{\frac{0.596953}{2}} = 0.741948 \tag{5-16}$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je izračunata P vrijednost ≥0.01 možemo zaključiti da je sekvenca slučajna. Velike vrijednosti $\chi 2$ (obs) (te zbog toga male P vrijednosti) bi označavale odstupanje raspodjele rang matrica od one koja odgovara slučajnom nizu.

Preporučena duljina unosa:

Vjerojatnosti za M = Q = 32 izračunate su i umetnute u testni kod. Moguće je odabrati M i Q različit od 32 ali tada će se morati izračunati vjerojatnost. Minimalan broj bitova koji se ispituju mora biti takav da je n ≥ 38MQ (tj. postoji najmanje 38 matrica). Za M = Q = 32, svaka sekvenca koja se ispituje trebala bi sadržavati najmanje 38.912 bita.

6) Test diskretne Fourier-ove transformacije *(eng. Discrete Fourier transform test)*

Svrha testiranja:

Promatrane karakteristike ovoga testu su amplitude diskretne Fourier-ove transformacije ulazne sekvence. Cilj testa je otkriti periodične pojave, odnosno otkriti ponavljajuće uzorke koji su međusobno previše blizu. Namjera je otkriti je li broj najviših vrhova amplituda koji prelaze prag od 95% značajno različit od 5%.

Poziv funkcije:

DiscreteFourierTransform(n), gdje je:

n – duljina niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$

Ispitivanje statistike testa i referentne distribucije:

d - normalizirana razlika između promatranog i očekivanog broja učestalih komponenti koje su iznad granice od 95%.

Referentna distribucija za statistiku testa je normalna distribucija.

Opis testa:

(1) Jedinice i nule ulazne sekvence (ε) zamjenjuju se vrijednostima -1 i 1, te se kreira sekvenca $X = x_1, x_2, \ldots x_n$ gdje je $x_i = 2\varepsilon_i - 1$

Na primjer, ako je n=10 i ε = 1001010011, tada je X=1,-1,-1,1,-1,1,-1,-1,1,1.

(2) Primjenjuje se diskretna Fourierova transformaciju (DFT) na X te se dobiva: S = DFT (X). S je niz složenih varijabli koji predstavljaju periodičke komponente sekvence bitova na različitim frekvencijama.

(3) Potrebno je izračunati $M = \mathrm{mod}(S') \equiv |S'|$ gdje je S' podniz koji se sastoji od prvih n72 elemenata od S, dok funkcija modula proizvodi sekvence vršnih vrijednosti.

(4) Potrebno je izračunati $T = \sqrt{\left(\log \frac{1}{0.05}\right) n}$, gdje T predstavlja prag vršnih vrijednosti. Pod pretpostavkom slučajnosti 95% vrijednosti dobivenih pomoću ovog testa ne smiju prijeći vrijednost T.

(5) Potrebno je izračunati $N_0 = \frac{0.95 \cdot n}{2}$, N_0 predstavlja očekivani teorijski (95%) broj vršnih vrijednosti (pod pretpostavkom slučajnosti) koji su manji od T. Za prethodni primjer $N_0 = 4.75$

(6) Potrebno je izračunati N_1 , N_1 predstavlja stvarni broj vršnih vrijednosti unutar M koji su manji od T.

Za prethodni primjer $N_1 = 4$

(7) Izračunati d

$$\mathrm{d} = \frac{(\mathrm{N_1} - \mathrm{N_0})}{\sqrt{\mathrm{n} \cdot 0.95 \cdot 0.05/4}} \tag{5-17}$$

Za prethodni primjer:

$$\mathrm{d} = \frac{(4 - 4.75)}{\sqrt{10 \cdot 0.95 \cdot 0.05/4}} = -2.176429. \tag{5-18}$$

(8) Potrebno je izračunati vrijednost P

$$\mathrm{P} = \mathrm{erfc}\left(\frac{|\mathrm{d}|}{\sqrt{2}}\right) \tag{5-19}$$

Za prethodni primjer:

$$\mathrm{P} = \mathrm{erfc}\left(\frac{|-2.176429|}{\sqrt{2}}\right) = 0.029523 \tag{5-20}$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je dobivena vrijednost (iz prethodnog primjera) P≥0.01 ulazna sekvenca može se smatrati slučajnom. Ukoliko je vrijednost d vrlo mala moguće je zaključiti da postoji premalo vršnih vrijednosti (<95%) ispod T i previše vršnih vrijednosti (>5%) iznad T.

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 1000 bitova.

7) Test ne preklapanja uzoraka *(eng.Non-overlapping Template Matching Test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je broj pojavljivanja unaprijed definiranih ciljeva pod niza. Svrha testa je odbaciti nizove koji pokazuju obilježja danog ne periodičnog uzorka. Prilikom pretraživanja niza koristi se m-bitni prozor koji pronalazi m-bitni uzorak. Ukoliko uzorak nije pronađen prozor se povećava za jedan bit, a ukoliko je uzorak pronađen prozor se resetira na prvi bit nakon pronađenog uzorka te se nastavlja pretraga.

Poziv funkcije:

NonOverlappingTemplateMatching(m,n), gdje je :

m- duljina bitova svakog predloška

n – duljina niza koji se testira

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

B - m-bitni predložak kojemu uzorak treba odgovarati, B je niz jedinica i nula (duljine m) koja je definirana u biblioteci predložaka ne-periodičnih uzoraka sadržanih unutar testnog koda.

M – duljina bitova podniza od ε koji će se testirati

N – broj neovisnih blokova, N je broj definiran unutar testnog koda i iznosi 8

Ispitivanje statistike testa i referentne distribucije:

$\chi 2$ (obs) – mjera koliko dobro se promatrani broj „pogodaka“ unutar predloška poklapa sa očekivanim brojem „pogodaka“ unutar predloška (pod pretpostavkom slučajnosti).

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Podijeliti ulaznu sekvencu u N neovisnih blokova duljine M

Na primjer, ako je ε= 101001001011110110, tada je n = 20. Ako je N=2 i M=3 tada postoje dva bloka, a to su 1010010010 i 1110010110.

(2) Neka je Wj (j = 1, ..., N) broj koji označava koliko puta se B (predložak) pojavljuje unutar bloka j. Potraga za pojavljivanjem počinje stvaranjem m-bitnog prozora na ulaznoj sekvenci, uspoređujući bitove unutar tog prozora sa bitovima prema predlošku. Ako se ne podudaranja, prozor se pomjera za jedan bit, npr., ako je m = 3, a trenutni prozor sadrži bitove od 3 do 5, sljedeći će se prozor sastojati od bitova od 4 do 6. Ako postoji podudaranje, prozor će se pomjeriti za m bitova, npr. ako trenutni (uspješni) prozor sadrži bitove od 3 do 5, sljedeći će se prozor sastojati od bitova od 6 do 8.

Za prethodni primjer, ako je m = 3, a predložak je B =001, tada se proces pretrage provodi na slijedeći način:

Tablica 5.3.5. Proces pretrage B

Pozicija bita	**Blok 1**		**Blok 2**	
	Bitovi	W_1	Bitovi	W_2
1-3	101	0	111	0
2-4	010	0	110	0
3-5	100	0	100	0
4-6	001 (pogodak)	Inkrement za 1	001 (pogodak)	Inkrement za 1
5-7	nije ispitan	-	Nije ispitan	-
6-8	Nije ispitan	-	Nije ispitan	-

7-9	001	Inkrement za 2	011	1
8-10	010 (pogodak)	2	110	1

Prema tablici 5.3.5, $W_1 = 2$, $W_2 = 1$.

(3) Pod pretpostavkom slučajnosti, potrebno je izračunati teorijsku srednju vrijednost μ i varijancu σ^2 :

$$\mu = \frac{(M - m + 1)}{2^m}$$

(5-21)

$$\sigma^2 = M\left(\frac{1}{2^m} - \frac{2 \cdot m - 1}{2^{2m}}\right)$$

(5-22)

Za prethodni primjer:

$$\mu = \frac{(10 - 3 + 1)}{2^3} = 1$$

(5-23)

$$\sigma^2 = 10\left(\frac{1}{2^3} - \frac{2 \cdot 3 - 1}{2^{2 \cdot 3}}\right) = 0.46875$$

(5-24)

(4) Potrebno je izračunati χ2 (obs)

$$\chi 2(\text{obs}) = \sum_{j=1}^{N} \frac{(W_j - \mu)^2}{\sigma^2}$$

(5-25)

Za prethodni primjer:

$$\chi 2(\text{obs}) = \frac{(2-1)^2 + (1-1)^2}{0.46875} = \frac{1+0}{0.46875} = 2.133333$$

(5-26)

(5) Potrebno je izračunati vrijednost P

$$P = igamc\left(\frac{N}{2}, \frac{\chi^2\,(obs)}{2}\right)$$

(5-27)

Pomoću ovog testa izračunat će se nekoliko P vrijednosti odnosno jedna vrijednost za svaki predložak. Za m=9 izračunat će se do 148 vrijednosti za P, dok za m=10 do 284 vrijednosti.

Za prethodni primjer:

$$P = igamc\left(\frac{2}{2}, \frac{2.133333}{2}\right) = 0.344154$$

(5-28)

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je izračunata vrijednost P ≥ 0.01 možemo zaključiti da je ulazna sekvenca slučajna te da prolazi ovaj test. Ukoliko je vrijednost P vrlo mala (<0.01) tada ulazna sekvenca ima nepravilne pojave bitova u odnosu na moguće uzorke predložaka.

Preporučena duljina unosa:

Testni kod napisan je na način da provede predloške za m=2,3,…,10. Preporuča se da je m=10 kako bi se dobili valjani rezultati. Iako je specificirano da je N=8, to je moguće promijeniti. N mora biti odabran na način da je N≤100 kako bi se dobili valjani rezultati. Također je potrebno ispuniti uvjet da je M>0.01·n i N=[n/M].

8) Test preklapanja uzoraka *(eng. Overlapping Template Matching Test)*

Svrha testiranja:

Karakteristika koja se promatra u ovom testu je broj pojavljivanja unaprijed definiranih ciljeva pod niza. Svrha testa je odbaciti nizove koji pokazuju odstupanje od očekivanog broja nizova jedinica zadane duljine. Pošto postoji odstupanje kod niza jedinica za očekivati je i odstupanje kod niza nula, međutim nizovi nula se ne promatraju odvojeno. Ovaj test ima vrlo sličan način rada kao i prethodni test, te također koristi prozore veličine m-bita za pretraživanje određenog m-bitnog uzorka.

Poziv funkcije:

OverlappingTemplateMatching(m,n), gdje je :

m – duljina bitova predloška - u ovom slučaju duljina ponavljanja jedinica

n – duljina testiranog niza

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

B – m-bitni predložak kojemu uzorak treba odgovarati

K – broj stupnjeva slobode, u ovom testnom kodu K je definiran i iznosi 5

M – duljina bitova podniza ε koji će se testirati, u ovom testnom kodu M je definiran i iznosi 1032

N – broj neovisnih blokova od n, u ovom testnom kodu N je definiran i iznosi 968

Ispitivanje statistike testa i referentna distribucija:

$\chi 2$ (obs) – mjera koliko dobro se promatrani broj „pogodaka“ unutar predloška poklapa sa očekivanim brojem „pogodaka“ unutar predloška (pod pretpostavkom slučajnosti).

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Podijeliti ulaznu sekvencu u N neovisnih blokova duljine M

Na primjer, ako je ε = 1011101111**0010110100**0111001011**1011111000**0101101001, tada je n = 50. Ako je K = 2, M = 10 i N = 5, tada su pet blokova slijedeći 1011101111, 0010110100, 0111001011, 1011111000 i 0101101001.

(2) Potrebno je izračunati broj pojavljivanja B u svakom od N blokova. Potraga za poklapanjem nastaje stvaranjem m-bitnog prozora na testiranoj sekvenci, uspoređujući bitove unutar tog prozora sa B te povećanjem brojača kada se dogodi poklapanje. Prozor se pomiče za jedan bit nakon svake pretrage. Npr. Ako je m=4 a prvi prozor sadrži bitove od 42 do 45 slijedeći prozor će se sastojati od bitova koji su na pozicijama od 43 do 46. Broj pojavljivanja B u svakom bloku se sprema na način da se niz v_i povećava (gdje je i=0,…5), tako da se v_0 povećava kada poklapanje sa B ne postoji, a v_1 kada prelapanje postoji. v_5 se povećava za 5 ili više podudaranja B sa testiranim podnizom (N).

Za prethodni primjer, ako je m = 2 i B = 11, pretraga prvog bloka (1011101111) odvija se na slijedeći način:

Tablica 5.3.6. Pretraga bitova B

Pozicija bita	***Bitovi***	***Broj pojavljivanja B = 11***
1-2	*10*	*0*
2-3	*01*	*0*
3-4	*11 (pogodak)*	*Inkrement za 1*
4-5	*11 (pogodak)*	*Inkrement za 2*
5-6	*10*	*2*
6-7	*01*	*2*
7-8	*11 (pogodak)*	*Inkrement za 3*
8-9	*11 (pogodak)*	*Inkrement za 4*
9-10	*11 (pogodak)*	*Inkrement za 5*

Nakon pretrage prvog bloka, B (11) pojavljuje se ukupno 5 puta, stoga je v_5 povećan, dok su vrijednosti $v_0 = v_1 = v_2 = v_3 = v_4 = 0$, a $v_5 = 1$

Na sličan način ispituju se preostali blokovi (od 2 do 5) . U drugom bloku 11 pojavljuje se 2 puta pa je v_2 povećava, u trećem bloku 11 pojavljuje se 3 puta (v_3 se povećava) u četvrtom bloku 11 pojavljuje se 4 puta (v_4 se povećava) dok se u petom bloku pojavljuje jednom (v_1 se povećava).

Nakon što su svi blokovi ispitani $v_0 = 0$, $v_1 = 1$, $v_2 = 1$, $v_3 = 1$, $v_4 = 1$, $v_5 = 1$.

(3) Potrebno je izračunati vrijednosti λ i η koje će se koristiti za računanje teorijske vrijednosti $л_i$

$$\lambda = \frac{(M - m + 1)}{2^m}$$

(5-29)

$$\eta = \frac{\lambda}{2}$$

(5-30)

Za prethodni primjer

$$\lambda = \frac{(10 - 2 + 1)}{2^2} = 2.25$$

(5-31)

$$\eta = \frac{2.25}{2} = 1.125$$

(5-32)

(4) Potrebno je izračunati $\chi 2(\text{obs})$

$$\chi 2(\text{obs}) = \sum_{i=0}^{5} \frac{(v_i - N\pi_i)^2}{N\pi_i}$$

(5-33)

gdje je π_0= 0.364091, π_1= 0.185659, π_2= 0.139381, π_3= 0.100571, π_4= 0.070432, π_5= 0.139865 .

Za prethodni primjer vrijednosti π_i moraju se ponovno računati budući da ulazne vrijednosti ne odgovaraju preporučenim. Vrijednosti su slijedeće π_0= 0.324652, π_1= 0.182617, π_2= 0.142670, π_3= 0.106645, π_4= 0.077147, π_5= 0.166269 .

$$\chi 2(\text{obs}) = \frac{(0-5\cdot 0.324652)^2}{5\cdot 0.324652} + \frac{(1-5\cdot 0.182617)^2}{5\cdot 0.182617} + \frac{(1-5\cdot 0.142670)^2}{5\cdot 0.142670} + \frac{(1-5\cdot 0.106645)^2}{5\cdot 0.106645} + \frac{(1-5\cdot 0.077147)^2}{5\cdot 0.077147} + \frac{(1-5\cdot 0.166269)^2}{5\cdot 0.166269} = 3.167729 \quad (5\text{-}34)$$

(5) Potrebno je izračunati vrijednost P

$$\mathrm{P} = \mathrm{igamc}\left(\frac{5}{2}, \frac{\chi 2(\text{obs})}{2}\right) \quad (5\text{-}35)$$

Za prethodni primjer:

$$\mathrm{P} = \mathrm{igamc}\left(\frac{5}{2}, \frac{3.167729}{2}\right) = 0.274932 \quad (5\text{-}36)$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je dobivena vrijednost (iz prethodnog primjera) P≥0.01 ulazna sekvenca može se smatrati slučajnom. Za 2-bitni predložak (B=11), ukoliko cijela sekvenca sadrži previše dvo-bitnih ponavljanja jedinica tada će v_5 biti prevelik, , a vrijednost P će biti mala (<0.01) što daje rezultat da sekvenca nije slučajna.

Preporučena duljina unosa:

Vrijednosti K,M i N odabrana su pod uvijetom da se svaka testirana sekvenca sastoji od minimalno 10^6 bitova. Različite vrijednsti m je moguće odabrati ali NIST preporuča da je m = 9 ili m=10. Ukoliko je odabrana neka druga vrijednost tada za ostale vrijednosti vrijedi:

- n≥MN
  N mora biti odabran tako da je $N\cdot(\min \pi_i) > 5$
- $\lambda = \frac{(M-m+1)}{2^m} \approx 2$
- m mora biti odabran tako da vrijedi $m \approx \log_2 M$
- K mora biti odabran tako da vrijedi $K \approx 2\lambda$, ukoliko je K≠5 tada je potrebno izračunati vrijednosti π_i

9) Maurer-ov univerzalni statistički test *(eng. Maurer's "Universal Statistical" Test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je broj bitova između odgovarajućih obrazaca. Cilj testa je otkriti da li je niz moguće značajno sažeti bez gubitka informacija. Niz koji je moguće znatno sažeti smatra se kao ne slučajnim nizom te se odbacuje.

Poziv funkcije:

Universal(L,Q,n), gdje je :

L – duljina bloka, upotreba oznake L kao duljine bloka nije u skladu sa do sada korištenom oznakom veličine bloka M , nego je navedena u izvornom kodu Maurerovog testa pa se iz tog razloga koristi

Q – broj blokova u inicijaliziranoj sekvenci

n – duljina sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

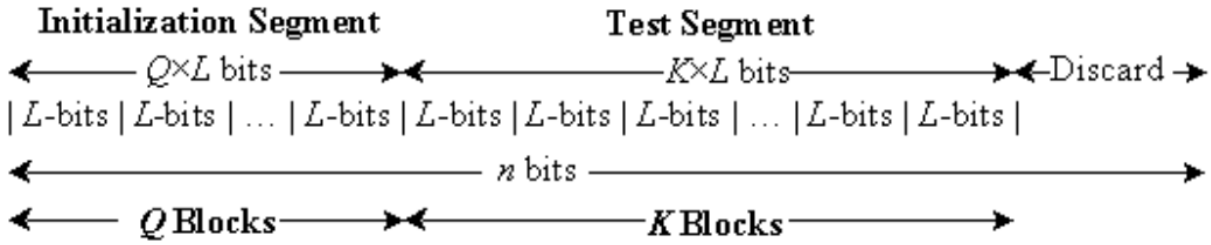
Ispitivanje statistike testa i referentna distribucija:

f_n - zbroj $\log_2$ udaljenosti između odgovarajućih L-bitnih predložaka, tj. Zbroj znamenaka između L-bitnih predložaka.

Referentna distribucija za statistiku testa je polu-normalna distribucija.

Opis testa:

(1) n-bitna sekvenca (ε) je podijeljena u dva segmenta: segment za inicijalizaciju koji se sastoji od Q L-bitnih blokova koji se ne preklapaju, i testni segment koji se sastoji od K L-bitnih blokova koji se također ne preklapaju. Preostali bitovi na kraju sekvence koji ne tvore potpuni L-bitni blok se odbacuje. Prvi Q blokovi koriste se za inicijaliziranje testa. Preostali K blokovi su testni blokovi ($K = \left[\frac{n}{L}\right] - Q$).



Slika 5.3.1 Prikaz podjele sekvence na inicijalizacijski i testni segment

Na primjer ako je ε = 01011010011101010111, tada je n = 20. Ako je L =2 i Q =4, tada je K = 6. Inicijalizacijski segment je 01011010, a testni 011101010111. L-bitni blokovi se prikazuju prema slijedećoj tablici.

Tablica 5.3.7. Prikaz L bitnih blokova

Blok	Tip	Sadržaj
1	*Inicijalizacijski segment*	01
2		01
3		10
4		10
5		01
6		11

7	*Testni segment*	01
8		01
9		01
10		11

(2) Korištenjem segmenta inicijalizacije kreira se tablica za svaku moguću L-bitnu vrijednost (tj., L-bitna vrijednost se koristi kao indeks u tablici). Broj bloka zadnje pojave svakog L-bitnog bloka označen je u tablici (tj. Za i od 1 do Q, Tj = i, gdje j predstavlja decimalni dio i-tog L-bitnog bloka).

Za prethodni primjer kreira se slijedeća tablica korištenjem četiri inicijalizacijska bloka.

Tablica 5.3.8. Mogući inicijalizacijski blokovi

	Moguća vrijednost L			
	00 (T_0)	01 (T_1)	10 (T_2)	11 (T_3)
Inicijalizacija	0	2	4	0

(3) Svaki K blok u testnom segmentu se provjerava te se određuje broj blokova od zadnje pojave istog L-bitnog bloka ($i - T_j$). Zatim se zamjenjuje vrijednost u tablici sa lokacijom trenutnog bloka ($T_j = i$). Dodaje se izračunata udaljenost između ponovnog pojavljivanja L-bitnog bloka na $\log_2$ zbroj svih razlika detektiranih u K blokovima (sum=sum + $\log_2$($i - T_j$).

Za prethodni primjer, tablica i kumulativni zbroj dobit će se na slijedeći način:

Za blok 5: (prvi testni blok): 5 se nalazi u "01" redu tablice (tj. T1),
sum = log2 (5-2) = 1.584962501.

Za blok 6: 6 se nalazi u "11" redu tablice (tj. T3),
sum = 1.584962501 + log2 (6-0) = 1.584962501 + 2.584962501 = 4.169925002.

Za blok 7: 7 se nalazi u "01" redu tablice (tj. T1),
sum = 4.169925002 + log2 (7-5) = 4.169925002 + 1 = 5.169925002.

Za blok 8: 8 se nalazi u "01" redu tablice (tj. T1),
sum = 5.169925002 + log2 (8-7) = 5.169925002 + 0 = 5.169925002.

Za blok 9: 9 se nalazi u "01" redu tablice (tj. T1),
sum = 5.169925002 + log2 (9-8) = 5.169925002 + 0 = 5.169925002.

Za blok 10: 10 se nalazi u "11" redu tablice (tj., T3),
sum = 5.169925002+ log2 (10-6) = 5.169925002 + 2 = 7.169925002.

Tablica 5.3.9. Prikaz mogućih stanja

Iteracijski blok	***Moguće vrijednosti L***			
	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) Potrebno je izračunati statistiku testa:

$$f_n = \frac{1}{K} \sum_{i=Q+1}^{Q+K} \log \log_2 (i - T_j)$$

(5-37)

Gdje je T_j podatak iz tablice koji odgovara decimalnom prikazu i-tog L-bitnog bloka

Za prethodni primjer

$$f_n = \frac{7.169925002}{6} = 1.1949875.$$

(5-38)

(5) Potrebno je izračunati vrijednost P

$$P = \text{erfc}\left(\left|\frac{f_n - \text{expectedValue (L)}}{\sqrt{2\sigma}}\right|\right) \tag{5-39}$$

Gdje je expectedValue (L) i σ očekivane vrijednosti preuzete iz tablice unaprijed izračunatih vrijednosti. Pod pretpostavkom slučajnosti očekivana vrijednost je teorijska očekivana vrijednost izračunate statističke vrijednosti za određenu L-bitnu duljinu. Teorijsko standardno odstupanje dano je izrazom :

$$\sigma = c\sqrt{\frac{variance\ (L)}{K}},\ gdje\ je\ c = 0.7 - \frac{0.8}{L} + \left(4 + \frac{32}{L}\right) \cdot \frac{K^{-\frac{3}{L}}}{15} \tag{5-40}$$

Tablica 5.3.10. Očekivana vrijednost izračunate statističke vrijednosti za određenu L-bitnu duljinu

L	***expectedValue***	***varijanca***
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***	***varijanca***
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*

Za prethodni primjer:

$$P = \text{erfc}\left(\left|\frac{1.1949875 - 1.5374383}{\sqrt{2} \cdot \sqrt{1.338}}\right|\right) = 0.767189 \tag{5-41}$$

ExpectedValue i varijanca za L=2 nisu prikazane u gornjoj tablici jer blok duljine L=2 nije preporučen za testiranje.

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Dobivena vrijednost P je ≥0.01, te s toga zaključujemo da je ulazna sekvenca slučajna. Ukoliko se f_n značajno razlikuje od expectedValue (L), slijed je moguće značajno kompresirati što bi značilo da ulazni niz ne zadovoljava uvijete slučajnosti. .

Preporučena duljina unosa:

Ovaj test zahtijeva dugi niz bitova (n ≥ (Q + K) L) koji su podijeljeni u dva segmenta L-bitnih blokova, gdje se L treba odabrati tako da 6 ≤ L ≤ 16. Prvi segment sastoji se od Q inicijalizacijskih blokova Q, gdje Q treba biti odabran tako da $Q = 10 \cdot 2^L$. Drugi segment sastoji se od K test blokova, gdje je $K = \left[\frac{n}{L}\right] - Q \approx 1000 \cdot 2^L$.

Vrijednosti L, Q i n bi se trebale izabrati na sljedeći način:

Tablica 5.3.11. Vrijednosti L i Q ovisne duljini sekvence n

n	**L**	$\boldsymbol{Q = 10 \cdot 2^L}$
≥ 387840	6	640
≥ 904960	7	1280
≥ 2068480	8	2560
≥ 4654080	9	5120
≥ 10342400	10	10240
≥ 22753280	11	20480
≥ 49643520	12	40960
≥ 107560960	13	81920
≥ 231669760	14	163840
≥ 496435200	15	327680
≥ 1059061760	16	655360

10) Test linearne složenosti *(eng. Linear Complexity Test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je duljina generiranog povratnog registra. Cilj je utvrditi dali je testirani niz dovoljno složen kako bi se mogao kvalificirati kao slučajan. Ispitivani niz dijeli se na M-bitne blokove te se računa minimalna veličina posmačnog registra s povratnom vezom koji generira sve bitove u tom bloku. Slučajno generirani nizovi imaju duže povratne registre, dok kratki povratni registri ukazuje na to da je slijed brojeva ili bitova nije slučajan.

Poziv funkcije:

LinearComplexity(M,n), gdje je:

M – duljina bloka

n – duljina sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

K – broj stupnjeva slobode, K je predefiniran u testnom kodu i iznosi 6

Ispitivanje statistike testa i referentna distribucija:

$\chi 2(\text{obs})$ - mjera koliko dobro promatrani broj pojavljivanja posmačnog registra fiksne duljine odgovara očekivanom broju pojavljivanja pod pretpostavkom slučajnosti.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) n – bitna sekvenca dijeli se u N neovisnih blokova po M bita gdje je n = MN

(2) Koristeći Berlekamp - Massey algoritam određuje se linearna složenost L_i svakog od N blokova (i = 1, ..., N). L_i je duljina najkraće sekvence posmačnog registra koji generira

sve bitove u bloku i. Bitovi unutar L_i-bitnog slijeda se zbrajaju te se na zbroj primjenjuje modulo 2 aritmetika kako bi se dobio slijedeći bit sekvence (bit L_i +1).

Na primjer, ako je M =13, a blok koji se testira 1101011110001, tada je L_i =4, novi niz dobiva se na način da se zbroje bitovi na poziciji 1 i 2 (unutar 4-bitnog pod niza) kako bi se dobio peti bit. Ostali bitovi dobit će se na sličan način prikazan u tablici 5.3.12:

Tablica 5.3.12. Prikaz procesa dobivanja petog bita 4-bitnog niza

	Bit 1	***Bit 2***	***Bit 3***	***Bit 4***	***Bit 5***
Prva 4 bita i rezultat za bit 5	1	1	0	1	0
Bitovi od 2 do 5 i rezultat za bit 6	1	0	1	0	1
Bitovi od 3 do 6 i rezultat za bit 7	0	1	0	1	1
Bitovi od 4 do 7 i rezultat za bit 8	1	0	1	1	1
Bitovi od 5 do 8 i rezultat za bit 9	0	1	1	1	1
Bitovi od 6 do 9 i rezultat za bit 10	1	1	1	1	0
Bitovi od 7 do 10 i rezultat za bit 11	1	1	1	0	0
Bitovi od 8 do 11 i rezultat za bit 12	1	1	0	0	0
Bitovi od 9 do 12 i rezultat za bit 13	1	0	0	0	1

Za testirani blok, odabrani algoritam s povratnom vezom funkcionira, ukoliko to nije slučaj moguće je koristiti druge algoritme (npr. Zbrajanje bitova na pozicijama 1 i 3 kako bi se dobio bit na poziciji 5 ili, zbrajanje bitova na pozicijama 1,2 i 3 kako bi se dobio bit na poziciji 6.

(3) Pod pretpostavkom slučajnosti potrebno je izračunati teorijsku srednju vrijednost μ

$$\mu = \frac{M}{2} + \frac{(9 + (-1)^{M+1})}{36} - \frac{\left(\frac{M}{3} + \frac{2}{9}\right)}{2^M} \tag{5-42}$$

Za prethodni primjer:

$$\mu = \frac{13}{2} + \frac{(9 + (-1)^{13+1})}{36} - \frac{\left(\frac{13}{3} + \frac{2}{9}\right)}{2^{13}} = 6.777222 \tag{5-43}$$

(4) Za svaki pod niz potrebno je izračunati vrijednost $T_{i,}$ gdje je:

$$T_i = (-1)^M \cdot (L_i - \mu) + \frac{2}{9} \tag{5-44}$$

Za prethodni primjer:

$$T_i = (-1)^{13} \cdot (4 - 6.777222) + \frac{2}{9} = 2.999444 \tag{5-45}$$

(5) Pomoću dobivenih T_i vrijednosti dobit će se vrijednosti $v_0,\ldots,v_6$ prema slijedećim uvjetima:

Ako je:	$T_i \leq -2.5$	*povećaj v_0 za 1*
	$-2.5 < T_i \leq -1.5$	*povećaj v_1 za 1*
	$-1.5 < T_i \leq -0.5$	*povećaj v_2 za 1*
	$-0.5 < T_i \leq 0.5$	*povećaj v_3 za 1*
	$0.5 < T_i \leq 1.5$	*povećaj v_4 za 1*
	$1.5 < T_i \leq 2.5$	*povećaj v_5 za 1*
	$T_i > 2.5$	*povećaj v_6 za 1*

(6) Potrebno je izračunati $\chi 2(\text{obs})$:

$$\chi 2(\text{obs}) = \sum_{i=0}^{K} \frac{(v_i - N\pi_i)^2}{N\pi_i} \tag{5-46}$$

gdje je $\pi_0 = 0.010417$, $\pi_1 = 0.03125$, $\pi_2 = 0.125$, $\pi_3 = 0.5$, $\pi_4 = 0{,}25$, $\pi_5 = 0{,}0625$, $\pi_6 = 0{,}020833$ su vjerojatnosti izračunate pomoću testnog koda.

(7) Potrebno je izračunati vrijednost P

$$P = igamc\left(\frac{K}{2}, \frac{\chi 2(\text{obs})}{2}\right) \tag{5-47}$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Ako je P vrijednost < 0.01 tada se promatrane brojčane vrijednosti T_i pohranjene u ν_i razlikuju od očekivanih vrijednosti. Očekuje se da će distribucija učestalosti T_i (u varijablama νi) biti proporcionalna izračunatom π_i kako je prikazano u koraku (6). Budući da podatci iz primjera ne zadovoljavaju minimalne uvijete za testiranje vrijednost P nije izračunata.

Preporučena duljina niza:

Minimalna duljina ulazne sekvence mora biti 10^6 ($n \geq 10^6$). Duljina bloka M mora biti unutar raspona $500 \leq M \leq 5000$, duljina neovisnih blokova N mora biti $N \geq 200$ kako bi rezultati za $\chi 2(\text{obs})$ bio valjan.

11) Serijski test *(eng. Serial test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je učestalost preklapanja svakog m-bitnog uzoraka unutar cijelog generiranog niza. Svrha testa je utvrditi dali je broj 2^m *m*-bitnih preklapajući uzoraka približno jednak onome koji se očekuje od pravog slučajnog niza.

Poziv funkcije:

Serial(m,n), gdje je:

m- duljina bloka

n- duljina sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

$\nabla\psi_m^2(obs) i\ \nabla^2\psi_m^2(obs)$ - Mjera koliko dobro se promatrani m-bitni uzorci odgovaraju očekivanim pojavljivanjima m-bitnih uzoraka.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Na ulaznu sekvencu ε dodaje se prefiks te se dobiva nova sekvenca ε' . Prefiks se dodaje na način da se prvih m-1 bita dodaje na kraj sekvence ε.

Na primjer, ako je n=10 i ε = 0011011101. Ako je m =3 tada je ε' = 001101110100. Ako je m = 2 tada je ε' = 00110111010. Ako je m = 1, tada je ε' = ε =0011011101.

(2) Određuje se učestalost svih mogućih preklapajućih m-bitnih blokova, svih mogućih preklapajućih (m-1) blokova i svih mogućih preklapajućih (m-2) blokova. Neka $v_{i_{1\ldots}i_m}$ označava pojavljivanje m-bitnih uzoraka $i_{1\ldots}i_m$, neka $v_{i_{1\ldots}i_{m-1}}$ označava pojavljivanje (m-1)-bitnih uzoraka $i_{1\ldots}i_{m-1}$ i neka $v_{i_{1\ldots}i_{m-2}}$ označava pojavljivanje (m-2)-bitnih uzoraka $i_{1\ldots}i_{m-2}$.

Za prethodni primjer, ako je m= 3, tada je (m-1)=2 i (m-2)=1. Učestalost svih 3-bitnih blokova je: v_{000} = 0, v_{010} =1, v_{011} =2, v_{100} =1, v_{101} =2, v_{110}=2, v_{111} =0. Učestalost svih (m-1)-bitnih blokova je: v_{00} =1, v_{01} =3, v_{10} =3, v_{11} =3. Učestalost svih (m-2)-bitnih blokova je: v_0 = 4, v_1 =6.

(3) Potrebno je izračunati ψ_m^2:

$$\psi_m^2 = \frac{2^m}{n}\sum_{i_{1\ldots}i_m}\left(v_{i_{1\ldots}i_m} - \frac{n}{2^m}\right)^2 = \frac{2^m}{n}\sum_{i_{1\ldots}i_m} v_{i_{1\ldots}i_m}^2 - n \tag{5-48}$$

$$\psi_{m-1}^2 = \frac{2^{m-1}}{n}\sum_{i_{1\ldots}i_{m-1}}\left(v_{i_{1\ldots}i_{m-1}} - \frac{n}{2^{m-1}}\right)^2 = \frac{2^{m-1}}{n}\sum_{i_{1\ldots}i_{m-1}} v_{i_{1\ldots}i_{m-1}}^2 - n \tag{5-49}$$

$$\psi_{m-2}^2 = \frac{2^{m-2}}{n}\sum_{i_{1\ldots}i_{m-2}}\left(v_{i_{1\ldots}i_{m-2}} - \frac{n}{2^{m-2}}\right)^2 = \frac{2^{m-2}}{n}\sum_{i_{1\ldots}i_{m-2}} v_{i_{1\ldots}i_{m-2}}^2 - n \tag{5-50}$$

Za prethodni primjer:

$$\psi_3^2 = \frac{2^3}{10}\,(0+1+1+4+1+4+4+1) - 10 = 12.8 - 10 = 2.8$$

$$\psi_2^2 = \frac{2^2}{10}(1+9+9+9) - 10 = 11.2 - 10 = 1.2$$

$$\psi_1^2 = \frac{2}{10}(1+36) - 10 = 10.4 - 10 = 0.4$$

(4) Potrebno je izračunati $\nabla\psi_{\mathrm{m}}^2$ i $\nabla^2\psi_{\mathrm{m}}^2$:

$$\nabla\psi_{\mathrm{m}}^2 = \psi_{\mathrm{m}}^2 - \psi_{\mathrm{m-1}}^2$$

$$\nabla^2\psi_{\mathrm{m}}^2 = \psi_{\mathrm{m}}^2 - 2\psi_{\mathrm{m-1}}^2 + \psi_{\mathrm{m-2}}^2$$

Za prethodni primjer:

$$\nabla\psi_{\mathrm{m}}^2 = \psi_{\mathrm{m}}^2 - \psi_{\mathrm{m-1}}^2 = \psi_3^2 - \psi_2^2 = 2.8 - 1.2 = 1.6$$

$$\nabla^2\psi_{\mathrm{m}}^2 = \psi_{\mathrm{m}}^2 - 2\psi_{\mathrm{m-1}}^2 + \psi_{\mathrm{m-2}}^2 = \psi_3^2 - 2\psi_2^2 + \psi_1^2 = 2.8 - 2\cdot(1.2) + 0.4 = 0.8$$

(5) Potrebno je izračunati vrijednosti P_1 i P_2:

$$P_1 = \mathrm{igamc}(2^{\mathrm{m-2}}, \nabla\psi_{\mathrm{m}}^2)$$

$$P_2 = \mathrm{igamc}(2^{\mathrm{m-3}}, \nabla^2\psi_{\mathrm{m}}^2)$$

Za prethodni primjer:

$$P_1 = \mathrm{igamc}(\,2, 1.6\,) = 0.808792$$

$$P_1 = \mathrm{igamc}(\,2, 0.8\,) = 0.670320$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da su vrijednosti P_1 *i* $P_2 \geq 0.01$ testirana sekvenca može se smatrati slučajnom. Ukoliko je vrijednost $\nabla\psi_m^2$ ili $\nabla^2\psi_m^2$ jako velika testirani m-bitni blokovi nisu ravnomjerni, odnosno ulazna sekvenca ne ispunjava minimalan uvjet za testiranje.

Preporučena duljina unosa:

Preporuča se odabrati m i n tako da vrijedi $m < [\log_2 n] - 2$.

12) Test aproksimativne entropije *(eng. Approximate Entropy Test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je učestalost svakog preklapanja *m*-bitnih uzoraka. Svrha testa je usporediti učestalost preklapanja blokova dva uzastopna bloka čija je duljina m i m+1 u odnosu na očekivani rezultat za niz slučajnih brojeva.

Poziv funkcije:

ApproximyteEntropy(m,n) gdje je:

m- duljina svakog bloka – u ovom slučaju prvi blok koji se koristi u ovom testu. m+1 je duljina drugog korištenog bloka

n- duljina sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

$\chi 2(\text{obs})$ - mjera koliko dobro promatrana vrijednost ApEn(m) odgovara očekivanoj vrijednosti.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Na ulaznu sekvencu ε dodaje se prefiks te se dobiva nova sekvenca koja se sastoji od ε i m-1 bita. Prefiks se dodaje na način da se prvih m-1 bita dodaje na kraj sekvence ε.

Na primjer ako je ε= 0100110101 i m=3, tada je n=10. Bitovi nula i jedan na početku sekvence dodaju se na kraj i dobiva se nova sekvenca za testiranje ε= 010011010101.

(2) Broj učestalosti pojavljivanja dobiva se pomoću n preklapajućih blokova (npr. ako se blok sastoji od ε_j do ε_{j+m-1} i ispitan u vremenu j, tada blok koji sadrži ε_{j+1} do ε_{j+m} je ispitan u vremenu j+1). Neka je broj mogućih m-bitnih (m+1) vrijednosti predstavljen kao C_i^m gdje je m m-bitna vrijednost.

Za prethodni primjer: preklapajući m-bitni blokovi (gdje je m = 3) postaju 010, 100, 001, 011, 110, 101, 010, 101, 010 i 101. Broj različitih pojavljivanja m-bitnih nizova je $2^m = 2^3 = 8$ i to su:

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

(3) Potrebno je izračunati vrijednost C_i^m za svaki i

$$C_i^m = \frac{\#i}{n} \tag{5-51}$$

Za prethodni primjer: $C_{000}^3 = 0, C_{001}^3 = 0.1\ ,\ C_{010}^3 = 0.1,\ C_{011}^3 = 0.1,$

$C_{100}^3 = 0.1,\ C_{101}^3 = 0.3, C_{110}^3 0.1\ , C_{111}^3 = 0$

(4) Potrebno je izračunati $\varphi^{(m)}$

$$\varphi^{(m)} = \sum_{i=0}^{2^m-1} \pi_i \log \pi_i \tag{5-52}$$

Gdje je $\pi_i = C_j^3 i\, j = \log_2 i$.

Za primjer u ovom dijelu, $\varphi^{(3)}$ = 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

(5) Koraci od 1 do 4 se ponavljaju, m se zamjenjuje sa m+1

Korak 1: Za primjer u ovom dijelu, m = 4, niz koji je potrebno testirati postaje 0100110101010.

Korak 2: Preklapajući blokovi postaju 0100, 1001, 0011, 0110, 1101, 1010, 0101, 1010, 0101, 1010. Izračunate vrijednosti su: # 0011 = 1, # 0100 = 1, # 0101 = 2, # 0110 = 1, # 1001 = 1, # 1010 = 3, # 1101 = 1, a sve druge kombinacije su jednake nuli.

Korak 3: $C_{0011}^4 = C_{0100}^4 = C_{0110}^4 = C_{1001}^4 = C_{1101}^4 = 0.1$, $C_{0101}^4 = 0.2$, $C_{1010}^4 = 0.3$, a sve ostale vrijednosti su nula.

Korak 4: $\varphi^{(4)}$ = 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) Potrebno je izračunati statistiku testa $\chi^2 = 2n[\log 2 - \mathrm{ApEn}(m)]$ (5-53)

Gdje je: $\mathrm{ApEn}(m) = \varphi^{(m)} - \varphi^{(m+1)}$

Za primjer u ovom dijelu: $\mathrm{ApEn}(3) = -1.643418 - (-1.834372) = 0.190954$

$$\chi^2 = 2 \cdot 10(0.693147 - 0.190954) = 0.502193$$

(7) Potrebno je izračunati vrijednost P = igamc $\left(2^{m-1}, \frac{\chi^2}{2}\right)$. (5-54)

Za primjer u ovom dijelu: $P = igamc\left(2^2, \frac{0.502193}{2}\right) = 0.261961.$ (5-55)

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je izračunata vrijednost P ≥ 0.01 (P= 0.261961), ulazna sekvenca može se smatrati slučajnom. Mala vrijednosti ApEn(m) indiciraju na to da unutar testirane sekvence postoje određene pravilnosti, dok velike vrijednosti ukazuju na nepravilnosti.

Preporučena duljina niza:

Preporuča se odabrati m i n tako da vrijedi $m < [\log_2 n] - 5$

13) Test kumulativne sume *(eng. Cumulative Sums Test)*

Svrha testiranja:

Karakteristika koja se promatra ovim testom je maksimalno odstupanje (od nule) od nasumičnog hoda koji je definiran uz pomoć kumulativni zbroj uz prilagođene vrijednosti (-1,+1) brojeva niza. je najveće odstupanje od nule kumulativne sume koja se dobije zbrajanjem članova modificiranog niza. Cilj testa je utvrditi dali je kumulativni zbroj djelomičnih nizova unutar ispitivanog niza premali ili prevelik u odnosu na očekivano ponašanje tog kumulativnog zbroja za slučajni niz. U ovom slučaju se kumulativni zbroj može smatrati slučajnim hodom. Slučajna digresija pravog slučajnog niza treba biti jako blizu nule.

Poziv funkcije:

CumulativeSums(mode,n) gdje je:

mode – prekidač za primjenu testa, mode = 0 testiranje ulazne sekvence sa desna na lijevo (naprijed) ili mode = 1 testiranje ulazne sekvence sa lijeva na desno (unatrag).

n- duljina ulazne sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda):

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentne distribucije:

Z – najveća digresija dobivena pomoću kumulativne sume odgovarajuće (-1,+1) sekvence.

Referentna distribucija za statistiku testa je normalna distribucija.

Opis testa:

(1) Formira se normalizirana sekvenca: jedinice i nule ulazne sekvence (ε) zamjenjuju se vrijednostima X_i , vrijednosti može biti -1 ili +1 koristeći jednadžbu $X_i = 2\varepsilon_i - 1$

Na primjer: ako je ε = 1011010111, tada je X = 1, (-1), 1, 1 (-1), 1, (-1), 1, 1, 1.

(2) Računa se djelomična suma S_i svih pod nizova, gdje svaki počinje sa X_1 (ukoliko je mode = 0) ili X_n (ukoliko je mode = 1).

Tablica 5.3.13. Prikaz računanja suma S_i ovisno o odabranom mode

Mode = 0	***Mode = 1***
$S_1 = X_1$	$S_1 = X_n$
$S_2 = X_1 + X_2$	$S_2 = X_n + X_{n-1}$
$S_3 = X_1 + X_2 + X_3$	$S_3 = X_n + X_{n-1} + X_{n-2}$
.	.
.	.
$S_k = X_1 + X_2 + X_3 + \ldots + X_k$	$S_k = X_n + X_{n-1} + X_{n-2} + \ldots + X_{n-k+1}$
.	.
.	.
$S_n = X_1 + X_2 + X_3 + \ldots + X_k + \ldots + X_n$	$S_n = X_n + X_{n-1} + X_{n-2} + \ldots + X_{n-k+1} + \ldots + X_1$

Odnosno, $S_k = S_{k-1} + X_k$ za mode = 0 i $S_k = S_{k-1} + X_{n-k+1}$ za mod = 1.

Za primjer u ovom dijelu, kada je mode = 0 i X = 1, (-1), 1, 1, (-1), 1, (-1), 1, 1, 1, tada su vrijednosti S slijedeće:

$S_1 = 1$

$S_2 = 1 + (-1) = 0$

$S_3 = 1 + (-1) + 1 = 1$

$S_4 = 1 + (-1) + 1 + 1 = 2$

$S_5 = 1 + (-1) + 1 + 1 + (-1) = 1$

$S_6 = 1 + (-1) + 1 + 1 + (-1) + 1 = 2$

$S_7 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) = 1$

$S_8 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 = 2$

$S_9 = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + 1 = 3$

$S_{10} = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + 1 + 1 = 4$

(3) Potrebno je izračunati statistiku testa $z = \max_{1 \le k \le n} |S_k|$, gdje je $\max_{1 \le k \le n} |S_k|$ najveća apsolutna vrijednost parcijalne sume S_k.

Za primjer u ovom dijelu najveća vrijednost $S_k = 4$ iz čega slijedi da je $z = 4$.

(4) Potrebno je izračunati vrijednost P

$$P = \sum_{k=\frac{\left(\frac{-n}{z}+1\right)}{4}}^{\frac{\left(\frac{n}{z}-1\right)}{4}} \left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k-1)z}{\sqrt{n}}\right)\right] +$$

$$\sum_{k=\frac{\left(\frac{-n}{z}-3\right)}{4}}^{\frac{\left(\frac{n}{z}-1\right)}{4}} \left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)\right] \quad (5\text{-}56)$$

Gdje je Φ - standardna normalna kumulativna funkcija raspodjele vjerojatnosti

Za primjer u ovom dijelu vrijednost P = 0.4116588.

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna.

Zaključak i interpretacija rezultata:

Budući da je izračunata vrijednost $P \geq 0.01$ ($P= 0.411658$), ulazna sekvenca može se smatrati slučajnom. Ukoliko je odabran mod = 0, velike vrijednosti statistike testa indiciraju da postoji previše jedinica ili previše nula na početnim dijelovima sekvence, a ukoliko je odabran mod = 1, tada postoji previše jedinica ili previše nula na krajnjim dijelovima testirane sekvence. Jako mala vrijednost statistike testa z ukazuje na to da su jedinice i nule unutar testirane sekvence izmiješane slično.

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 100 bitova.

14) Test slučajne digresije *(eng. Random excursion test)*

Svrha testiranja:

Promatrana karakteristika ovog niza je broj ciklusa koji ima točno K obilazaka u kumulativnom zbroju slučajnih hodova. Kumulativna suma slučajnih brojeva hoda nalazi se unutar u parcijalne sume (0,1) sekvence koja se prilagođava za (-1, +1). Cilj ovog testa je odrediti broj posjeta u stanje unutar slučajnog hoda koji prekoračuju ono što se očekuje od slučajne sekvence.

Poziv funkcije:

RandomExcursion(n) gdje je :

n- duljina sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon= \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentna distribucija:

$\chi 2$(obs) - za određeno stanje x, mjera koliko dobro promatrani broj „posjeta“ stanja bita (0,1) unutar ciklusa odgovara očekivanom broju „posjeta“ stanja bita unutar ciklusa, pod pretpostavkom slučajnosti.

Referentna distribucija za statistiku testa je $\chi 2$ distribucija.

Opis testa:

(1) Formira se normalizirana sekvenca X : jedinice i nule ulazne sekvence (ε) zamjenjuju se sa - 1 i +1 koristeći jednadžbu $X_i = 2\varepsilon_i - 1$

Na primjer, ako je ε = 0110110101, tada je n = 10 i X = -1, 1, 1, -1, 1, 1, -1, 1, -1, 1.

(2) Računa se dijelomična suma S_i podnizova, gdje svaki počinje sa X_1.
Formira se set S= {S_i}

$S_1 = X_1$
$S_2 = X_1 + X_2$
$S_3 = X_1 + X_2 + X_3$
.
.
$S_k = X_1 + X_2 + X_3 + \ldots + X_k$
.
.
$S_n = X_1 + X_2 + X_3 + \ldots + X_k + \ldots + X_n$

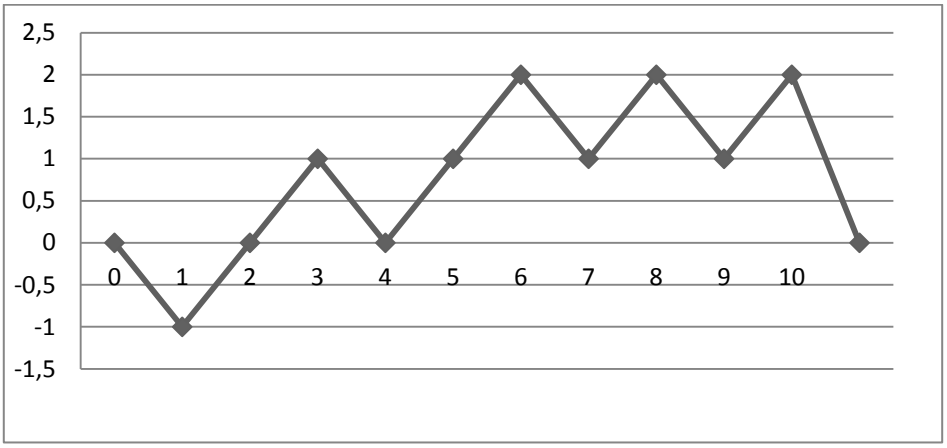
Za primjer u ovom dijelu:

$S_1 = -1$ $S_6 = 2$
$S_2 = 0$ $S_7 = 1$
$S_3 = 1$ $S_8 = 2$
$S_4 = 0$ $S_9 = 1$
$S_5 = 1$ $S_{10} = 2$

Set S = {-1, 0, 1, 0, 1, 2, 1, 2, 1, 2}.

(3) Formira se nova sekvenca S' na način da se na set S (sekvence) na početku i na kraju doda nula (S' =0, S_1,S_2, … S_n ,0).

Za primjer u ovom dijelu: S' = 0, -1, 0, 1, 0, 1, 2, 1, 2, 1, 2,0. Pomoću sekvence S' grafički možemo prikazati slučajan hod.



Slika 5.3.2. Prikaz slučajnog hoda sekvence S'

(4) Neka je J ukupan broj križanja nule unutar S', gdje je križanje nule pojava nula u S' koje se pojavljuju nakon početne nule. J je također i broj ciklusa S', gdje je ciklus S' pod niz niza S' koji se sastoji od pojavljivanja nule iza koje je ne nulta vrijednost te završava sa još jednom nulom. Nula na kraju ciklusa može biti početna nula drugog ciklusa. Broj ciklusa u S' je broj križanja nula. Ako je J<500 test se prekida.

Za primjer u ovom dijelu: ako je S'= {0, –1, 0 1, 0, 1, 2, 1, 2, 1, 2, 0}, tada je J=3 (postoje nule na pozicijama 3,5 i 12). Budući da je J=3 postoje ukupno 3 ciklusa a to su {0, -1, 0}, {0, 1, 0} i {0, 1, 2, 1, 2, 1, 2, 0}. Križanje nule vrlo lako je uočiti iz prikazanog grafa.

(5) Za svaki ciklus i za svaku ne nultu vrijednost x koja ima vrijednosti $-4 \leq x \leq -1$ i $1 \leq x \leq 4$ potrebno je izračunati učestalost pojavljivanja x.

Za primjer u ovom dijelu: prvi ciklus ima jednu pojavu od -1, drugi ciklus ima jednu pojavu od 1, a treći ciklus ima tri pojave 1 i tri pojave 2. Pojavljivanje x može se prikazati slijedećom tablicom.

Tablica 5.3.14. Pojavljivanje stanja x u ciklusima

Stanje x	**Ciklusi**		
	Ciklus 1 (0,-1,0)	**Ciklus 2 (0,1,0)**	**Ciklus 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) Za svaku od osam mogućih stanja x, potrebno je izračunati $v_k(x)$ gdje je v_k ukupan broj ciklusa u kojima se stanje x pojavljuje točno k puta kroz sve cikluse, za k =0,1,..,5 (ako je k=5 sva pojavljivanja koja su ≥ 5 pohranit će se u $v_5(x)$. $\sum_{k=0}^{5} v_k(x) = J$.

Za primjer u ovom dijelu:

$v_0(-1) = 2$ (stanje -1 pojavljuje se točno nula puta u dva ciklusa),
$v_1(-1) = 1$ (stanje -1 pojavljuje se samo jednom u jednom ciklusu)
$v_2(-1) = v_3(-1) = v_4(-1) = v_5(-1) = 0$ (stanje -1 se pojavljuje se točno {2, 3, 4, ≥5} puta u nula ciklusa).

$v_0(1) = 1$ (stanje 1 pojavljuje se točno 0 puta u 1 ciklusu)
$v_1(1) = 1$ (stanje 1 pojavljuje se samo jednom u 1 ciklusu)
$v_3(1) = 1$ (stanje 1 pojavljuje se točno tri puta u jednom ciklusu
$v_2(1) = v_2(1) = v_5(1) = 0$ (stanje 1 pojavljuje se točno {0, 4,> 5} puta u niti jednom ciklusu).

I tako dalje ….

Ovaj proces može se prikazati slijedećom tablicom:

Tablica 5.3.15. Proces dobivanja pojavljivanja stanja x u svakom ciklusu

Stanje x	**Broj ciklusa**					
	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) Za svih mogućih 8 stanja x potrebno je izračunati statistiku testa $\chi 2(\text{obs})$

$$\chi 2(\text{obs}) = \sum_{k=0}^{5} \frac{\left(v_k(x) - J\pi_k(x)\right)^2}{J\pi_k(x)} \tag{5-57}$$

Gdje je $\pi_k(x)$ vjerojatnost da se stanje x pojavljuje k puta u nasumičnoj distribuciji

Ukupno će se izračunati osam statistika $\chi 2$ (tj. Za x = -4, -3, -2, -1, 1, 2, 3, 4).

Za primjer u ovom dijelu:

$$\chi 2 = \frac{\left(1 - 3(0.5)\right)^2}{3(0.5)} + \frac{\left(1 - 3(0.25)\right)^2}{3(0.25)} + \frac{\left(0 - 3(0.125)\right)^2}{3(0.125)} + \frac{\left(1 - 3(0.0625)\right)^2}{3(0.0625)} + \frac{\left(0 - 3(0.0312)\right)^2}{3(0.0312)} + \frac{\left(0 - 3(0.0312)\right)^2}{3(0.0312)} = 4.333033 \tag{5-58}$$

(8) Za svako stanje x potrebno je izračunati vrijednost P

$$\mathrm{P} = \mathrm{igamc}\left(\frac{5}{2}, \frac{\chi 2(\mathrm{obs})}{2}\right) \tag{5-59}$$

Za primjer u ovom dijelu:

$$P = igamc\left(\frac{5}{2}, \frac{4.3330033}{2}\right) = 0.502529 \tag{5-60}$$

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna

Zaključak i interpretacija rezultata:

Budući da je izračunata vrijednost P ≥ 0.01 (P= 0.502529), ulazna sekvenca može se smatrati slučajnom. Ako su vrijednosti $\chi 2$ (obs) prevelike, tada ulazna sekvenca ima veliko odstupanje od teorijske distribucije za određeno stanje u svim ciklusima.

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 100000 bitova.

15) Test slučajne digresije varijanta *(eng. Random Excursions Variant Test)*

Svrha testiranja:

Promatrana karakteristika ovog testa je broj pojavljivanja određenog stanja u kumulativnog sumi slučajnog hoda. Svrha testa je otkriti odstupanja od očekivanog broja pojavljivanja različitih stanja u slučajnom nizu.

Poziv funkcije:

RandomExcursionVariant(n), gdje je:

n – duljina ulazne sekvence

Dodatan ulaz koji koristi funkcija (omogućen pomoću testnog koda) :

ε - sekvenca bitova generiranih pomoću slučajnog ili pseudoslučajnog generatora koji se testira, $\varepsilon = \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_n$.

Ispitivanje statistike testa i referentne distribucija:

ξ - za dano stanje x, ukupan broj pojavljivanja stanja kroz cijeli slučajni hod.

Referentna statistika testa je polu normalna (za veliki n). Ako sekvenca slučajna statistika testa će biti vrlo blizu nule, a ukoliko se sekvenca sastoji od previše nula ili previše jedinica statistika će biti jako velika.

Opis testa:

(1) Formira se normalizirana sekvenca X : jedinice i nule ulazne sekvence (ε) zamjenjuju se sa - 1 i +1 koristeći jednadžbu $X_i = 2\varepsilon_i - 1$

Na primjer ako je ε = 0110110101, tada je n = 10 i X = -1, 1, 1, -1, 1, 1, -1, 1, -1, 1.

(2) Računa se djelomična suma S_i pod nizova, gdje svaki počinje sa X_1.
Formira se set $S = \{S_i\}$

$S_1 = X_1$
$S_2 = X_1 + X_2$
$S_3 = X_1 + X_2 + X_3$
.
.
$S_k = X_1 + X_2 + X_3 + \ldots + X_k$
.
.
$S_n = X_1 + X_2 + X_3 + \ldots + X_k + \ldots + X_n$

Za primjer u ovom dijelu:

$S_1 = -1$ $S_6 = 2$

$S_2 = 0$ $S_7 = 1$

$S_3 = 1$ $S_8 = 2$

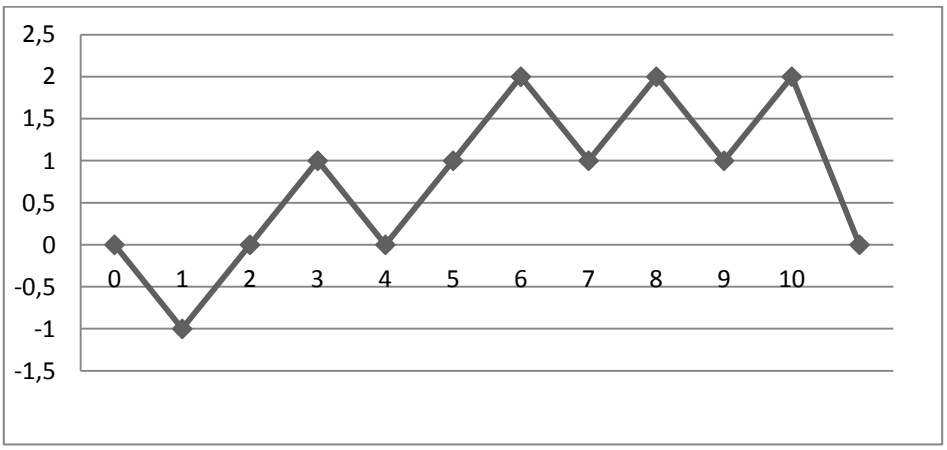
$S_4 = 0$ $S_9 = 1$

$S_5 = 1$ $S_{10} = 2$

Set S = {-1, 0, 1, 0, 1, 2, 1, 2, 1, 2}.

(3) Formira se nova sekvenca S' na način da se na set S (sekvence) na početku i na kraju doda nula (S' =0, S_1,S_2, … S_n ,0).

Za primjer u ovom dijelu: S' = 0, -1, 0, 1, 0, 1, 2, 1, 2, 1, 2,0. Pomoću sekvence S' grafički možemo prikazati slučajan hod.



Slika 5.3.3. Prikaz slučajnog hoda sekvence S'

(4) Za svako od osamnaest mogući stanja x potrebno je izračunati ξ (x) (ukupan broj pojavljivanja stanja x kroz sve J cikluse)

Za primjer u ovom dijelu:

$\xi(-1) = 1$, $\xi(1) = 4$, $\xi(2) = 3$, i svi ostali $\xi(x) = 0$.

(5) Za svaki ξ(x) potrebno je izračunati vrijednost P (ukupno će se izračunati 18 vrijednosti)

$$P = erfc\left(\frac{|\xi(x) - J|}{\sqrt{2J(4|x| - 2)}}\right)$$

(5-61)

Za primjer u ovom dijelu: x=1

$$P = erfc\left(\frac{|4-3|}{\sqrt{2 \cdot 3(4-2)}}\right) = 0.683091$$

(5-62)

Pravilo odlučivanja (na razini od 1%):

Ako je izračunata vrijednost P <0.01 možemo zaključiti da ulazna sekvenca nije slučajna. U suprotnom sekvenca se može smatrati kao slučajna

Zaključak i interpretacija rezultata:

Budući da je izračunata vrijednost P ≥ 0.01 (P= 0.6983091), ulazna sekvenca može se smatrati slučajnom. Izračunata je vrijednost P za stanje x=1, potrebno je izračunati i ostalih 17 stanja kako bi se dobio konačan rezultat (svih osamnaest vrijednosti P moraju biti ≥ 0.01 kako bi se moglo zaključiti da je ulazna sekvenca slučajna) .

Preporučena duljina unosa:

Preporuka je da se svaka ulazna sekvenca koju je potrebno testirati sastoji od minimalno 100000 bitova.

6. TESTIRANJE SLUČAJNOSTI BINARNIH SEKVENCI

U praktičnom dijelu diplomskog rada prikazat će se rezultati testiranja binarnih sekvenci generiranih pomoću jednog od mnogih online generatora pseudo slučajnih brojeva. Kako bi se utvrdilo dali je generirani niz zaista slučajan ili ne nizovi će testirati pomoću tri različita seta testova. Setovi testova za testiranje slučajnosti su ENT, DIEHARD i NIST. Svi testovi koji se nalaze unutar svakog seta testa opisani su u prethodnim poglavljima. Postoji jako puno implementacija statističkih testova u različitim programskim jezicima koje je moguće pokrenuti na različitim operacijskim sustavima. Sekvence testirane setovima DIEHARD i NIST testirane su pomoću VirtualBox-a na operacijskom sustavu linux, dok je set ENT testiran pomoću operacijskog sustava Windows 7. Ukupno je testirano deset različitih sekvenci prikazanih u tablici 6.1.

Tablica 6.1. Testirane sekvence

Naziv sekvence	Duljina sekvence (bit)	Veličina sekvence (MB)	Testirana setom
Sekvenca1N	8388608	1	NIST
Sekvenca2N	16777216	2	NIST
Sekvenca5N	41943040	5	NIST
Sekvenca10N	83886080	10	NIST
Sekvenca20N	167772160	20	NIST
SekvencaTN	128000	0.0152	NIST
SekvencaT2N	15360	0.0018	NIST
Sekvenca1ED	575472	0.068601	ENT, DIEHARD
Sekvenca2ED	575768	0.068636	ENT, DIEHARD
Sekvenca3ED	576016	0.068666	ENT, DIEHARD

6.1. Prikaz rezultata testiranja

Tablica 6.1.1. Rezultati testiranja Sekvenca1N

Test	P-Vrijednost	Zaključak
01. Frequency Test (Monobit)	0.5300039399876912	Slučajan
02. Frequency Test within a Block	0.08725670001215594	Slučajan

03. Run Test		0.9111360840211135	Slučajan
04. Longest Run of Ones in a Block		0.42288121212135976	Slučajan
05. Binary Matrix Rank Test		0.5481309246601644	Slučajan
06. Discrete Fourier Transform (Spectral) Test		0.2997549107824605	Slučajan
07. Non-Overlapping Template Matching Test		0.45747215701029353	Slučajan
08. Overlapping Template Matching Test		0.7756318059593126	Slučajan
09. Maurer's Universal Statistical test		0.01487355902271136	Slučajan
10. Linear Complexity Test		0.24194653626702955	Slučajan
11. Serial test:		0.8681283608171806	Slučajan
		0. 9367937720991389	Slučajan
12. Approximate Entropy Test		0.45008978769006525	Slučajan
13. Cummulative Sums (Forward) Test		0.5868320457034435	Slučajan
Cummulative Sums (Reverse) Test		0.6848355945934672	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	4.665144003618564	0.4580962011883827	Slučajan
-3	7.501513978494626	0.18593260423152413	Slučajan
-2	12.50033187309173	0.028539358386896627	Slučajan
-1	9.637992831541219	0.08616694371455409	Slučajan
+1	1.6379928315412184	0.8966171316958587	Slučajan
+2	0.9555732554537811	0.9660689167713238	Slučajan
+3	1.2859010752688171	0.9363757395241299	Slučajan
+4	0.47549184255664084	0.9929932162455853	Slučajan
15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	874	0.21410761757134056	Slučajan
-8.0	836	0.12595231493241338	Slučajan
-7.0	841	0.1064384567768446	Slučajan
-6.0	878	0.12878385025761666	Slučajan
-5.0	883	0.10018733890740317	Slučajan
-4.0	888	0.06814324816908449	Slučajan
-3.0	911	0.05231455106271625	Slučajan
-2.0	934	0.0261384942891572	Slučajan

-1.0	1048	0.15005569529769452	Slučajan
+1.0	1138	0.6414537821616817	Slučajan
+2.0	1193	0.34671221061859003	Slučajan
+3.0	1202	0.41559990784068024	Slučajan
+4.0	1150	0.7856153264523985	Slučajan
+5.0	1102	0.921314634524165	Slučajan
+6.0	1112	0.9796338261293225	Slučajan
+7.0	1139	0.8925933670663179	Slučajan
+8.0	1213	0.59602465391439	Slučajan
+9.0	1327	0.2787171714459288	Slučajan

U tablici 6.1.1. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. Iz rezultata je vidljivo da sekvenca prolazi svih 15 testova. Sekvenca1N sastoji se od 8388608 bitova. Vrijednost P svakog od 15 testova zadovoljava uvjet $P \geq 0.01$, iz tog razloga možemo zaključiti da je ulazna sekvenca generirana pomoću pseudoslučajnog generatora uistinu slučajna.

Tablica 6.1.2. Rezultati testiranja Sekvenca2N

Test	**P-Vrijednost**	**Zaključak**
01. Frequency Test (Monobit)	0.20551308225426745	Slučajan
02. Frequency Test within a Block	0.9911360703270604	Slučajan
03. Run Test	0.02797717544600238	Slučajan
04. Longest Run of Ones in a Block	0.6363879533297785	Slučajan
05. Binary Matrix Rank Test	0.8882216261558915	Slučajan
06. Discrete Fourier Transform (Spectral) Test	0.10629190635090904	Slučajan
07. Non-Overlapping Template Matching Test	0.7467909694763616	Slučajan
08. Overlapping Template Matching Test	0.4191173111747315	Slučajan
09. Maurer's Universal Statistical test	0.810329420982763	Slučajan
10. Linear Complexity Test	0.2555783741563739	Slučajan
11. Serial test:	0.437657930075819	Slučajan
	0.5223555712022698	Slučajan
12. Approximate Entropy Test	0.02511113375415918	Slučajan
13. Cummulative Sums (Forward) Test	0.3302115462120518	Slučajan

Cummulative Sums (Reverse) Test		0.34946879755628535	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	2.568877709262478	0.76608809628078	Slučajan
-3	3.002694224924011	0.6995704394749986	Slučajan
-2	3.042027843446283	0.6935062848274391	Slučajan
-1	3.5045592705167175	0.6226978645873046	Slučajan
+1	3.5592705167173255	0.6144388357451331	Slučajan
+2	1.8544410672070248	0.868896942936863	Slučajan
+3	1.8733665653495448	0.8663750444018538	Slučajan
+4	8.978719606445644	0.10991597767001794	Slučajan
15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	215	0.2810906017110435	Slučajan
-8.0	234	0.33895308347475606	Slučajan
-7.0	266	0.495763517547109	Slučajan
-6.0	257	0.3973866401214917	Slučajan
-5.0	255	0.33624716284630174	Slučajan
-4.0	271	0.3927692076799011	Slučajan
-3.0	299	0.600956137252728	Slučajan
-2.0	306	0.6046874812397751	Slučajan
-1.0	294	0.17242847881363021	Slučajan
+1.0	364	0.17242847881363021	Slučajan
+2.0	367	0.3923942974889111	Slučajan
+3.0	358	0.6131432169689941	Slučajan
+4.0	316	0.8480949776720867	Slučajan
+5.0	284	0.5587085108431276	Slučajan
+6.0	302	0.7509691868571475	Slučajan
+7.0	336	0.9396692106349527	Slučajan
+8.0	359	0.7626755265241323	Slučajan
+9.0	369	0.705281340218586	Slučajan

U tablici 6.1.2. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. Iz rezultata je vidljivo da sekvenca prolazi svih 15 testova. Sekvenca2N sastoji se od 16777216 bitova. Vrijednost P svakog od 15 testova zadovoljava uvjet $P \geq 0.01$, iz tog razloga možemo zaključiti da je ulazna sekvenca generirana pomoću pseudoslučajnog generatora uistinu slučajna.

Tablica 6.1.3. Rezultati testiranja Sekvenca5N

Test		**P-Vrijednost**	**Zaključak**
01. Frequency Test (Monobit)		0.5300039399876912	Slučajan
02. Frequency Test within a Block		0.08725670001215594	Slučajan
03. Run Test		0.9111360840211135	Slučajan
04. Longest Run of Ones in a Block		0.42288121212135976	Slučajan
05. Binary Matrix Rank Test		0.5481309246601644	Slučajan
06. Discrete Fourier Transform (Spectral) Test		0.2997549107824605	Slučajan
07. Non-Overlapping Template Matching Test		0.45747215701029353	Slučajan
08. Overlapping Template Matching Test		0.7756318059593126	Slučajan
09. Maurer's Universal Statistical test		0.01487355902271136	Slučajan
10. Linear Complexity Test		0.24194653626702955	Slučajan
11. Serial test:		0.8681283608171806	Slučajan
		0. 9367937720991389	Slučajan
12. Approximate Entropy Test		0.45008978769006525	Slučajan
13. Cummulative Sums (Forward) Test		0.5868320457034435	Slučajan
Cummulative Sums (Reverse) Test		0.6848355945934672	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	4.665144003618564	0.4580962011883827	Slučajan
-3	7.501513978494626	0.18593260423152413	Slučajan
-2	12.50033187309173	0.028539358386896627	Slučajan
-1	9.637992831541219	0.08616694371455409	Slučajan
+1	1.6379928315412184	0.8966171316958587	Slučajan
+2	0.9555732554537811	0.9660689167713238	Slučajan
+3	1.2859010752688171	0.9363757395241299	Slučajan
+4	0.47549184255664084	0.9929932162455853	Slučajan
15. Random Excursions Variant Test:			

Stanje	Ukupan broj	P-Vrijednost	Zaključak
-9.0	874	0.21410761757134056	Slučajan
-8.0	836	0.12595231493241338	Slučajan
-7.0	841	0.1064384567768446	Slučajan
-6.0	878	0.12878385025761666	Slučajan
-5.0	883	0.10018733890740317	Slučajan
-4.0	888	0.06814324816908449	Slučajan
-3.0	911	0.05231455106271625	Slučajan
-2.0	934	0.0261384942891572	Slučajan
-1.0	1048	0.15005569529769452	Slučajan
+1.0	1138	0.6414537821616817	Slučajan
+2.0	1193	0.34671221061859003	Slučajan
+3.0	1202	0.41559990784068024	Slučajan
+4.0	1150	0.7856153264523985	Slučajan
+5.0	1102	0.921314634524165	Slučajan
+6.0	1112	0.9796338261293225	Slučajan
+7.0	1139	0.8925933670663179	Slučajan
+8.0	1213	0.59602465391439	Slučajan
+9.0	1327	0.2787171714459288	Slučajan

U tablici 6.1.3. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. Iz rezultata je vidljivo da sekvenca prolazi svih 15 testova. Sekvenca5N sastoji se od 41943040 bitova. Vrijednost P svakog od 15 testova zadovoljava uvjet $P \geq 0.01$, iz tog razloga možemo zaključiti da je ulazna sekvenca generirana pomoću pseudoslučajnog generatora uistinu slučajna.

Tablica 6.1.4. Rezultati testiranja Sekvenca10N

Test	P-Vrijednost	Zaključak
01. Frequency Test (Monobit)	0.22781749043020916	Slučajan
02. Frequency Test within a Block	0.8474009742081183	Slučajan
03. Run Test	0.6778088214999062	Slučajan
04. Longest Run of Ones in a Block	0.8948356852098623	Slučajan
05. Binary Matrix Rank Test	0.22457224293132408	Slučajan

06. Discrete Fourier Transform (Spectral) Test		0.11238657565722307	Slučajan
07. Non-Overlapping Template Matching Test		0.8775158158618096	Slučajan
08. Overlapping Template Matching Test		0.5240341500597878	Slučajan
09. Maurer's Universal Statistical test		0.42020517362041376	Slučajan
10. Linear Complexity Test		0.4833912239347471	Slučajan
11. Serial test:		0. 5557872384422028	Slučajan
		0. 4026190879806535	Slučajan
12. Approximate Entropy Test		0. 311856334911603	Slučajan
13. Cummulative Sums (Forward) Test		0. 3761904718231919	Slučajan
Cummulative Sums (Reverse) Test		0. 22366592555949663	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	14.678031736140476	0.011830398072501315	Slučajan
-3	0.9986961832061082	0.9626708665079423	Slučajan
-2	1.3223541607765528	0.9326147295288569	Slučajan
-1	4.400763358778626	0.49326971759723337	Slučajan
+1	2.480916030534351	0.7793663723042163	Slučajan
+2	4.8762133634907165	0.4311734674894472	Slučajan
+3	4.748925190839698	0.4472835446549762	Slučajan
+4	2.828325347898935	0.7264323959338369	Slučajan
15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	770	0.14082242467396408	Slučajan
-8.0	781	0.13211611954341082	Slučajan
-7.0	761	0.08209464318526433	Slučajan
-6.0	773	0.07012648267467876	Slučajan
-5.0	831	0.11411873672866904	Slučajan
-4.0	912	0.2615314014423077	Slučajan
-3.0	983	0.5254675213700046	Slučajan
-2.0	994	0.4958810339380927	Slučajan
-1.0	1014	0.45769454835738277	Slučajan
+1.0	1018	0.512288755270045	Slučajan
+2.0	1006	0.596351197819641	Slučajan

+3.0	1065	0.8681087555165277	Slučajan
+4.0	1120	0.552236338243371	Slučajan
+5.0	1126	0.5700976861069749	Slučajan
+6.0	1156	0.4769196957801617	Slučajan
+7.0	1143	0.5649433691998899	Slučajan
+8.0	1092	0.8040196202676292	Slučajan
+9.0	1109	0.7465785038707997	Slučajan

U tablici 6.1.4. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. Iz rezultata je vidljivo da sekvenca prolazi svih 15 testova. Sekvenca10N sastoji se od 83886080 bitova. Vrijednost P svakog od 15 testova zadovoljava uvjet $P \geq 0.01$, iz tog razloga možemo zaključiti da je ulazna sekvenca generirana pomoću pseudoslučajnog generatora uistinu slučajna.

Tablica 6.1.5. Tablica 6.1.4. Rezultati testiranja Sekvenca20N

Test		**P-Vrijednost**	**Zaključak**
01. Frequency Test (Monobit)		0.3864901617883857	Slučajan
02. Frequency Test within a Block		0.20793263272685697	Slučajan
03. Run Test		0.8707152160090074	Slučajan
04. Longest Run of Ones in a Block		0.338250450280637	Slučajan
05. Binary Matrix Rank Test		0.15223115128078554	Slučajan
06. Discrete Fourier Transform (Spectral) Test		0.8114216873980027	Slučajan
07. Non-Overlapping Template Matching Test		0.9352128971769889	Slučajan
08. Overlapping Template Matching Test		0.8622973366009516	Slučajan
09. Maurer's Universal Statistical test		0.19859868219441512	Slučajan
10. Linear Complexity Test		0.4123077173827377	Slučajan
11. Serial test:		0. 6377831903514938	Slučajan
		0. 47313700716237	Slučajan
12. Approximate Entropy Test		0. 9023999178390809	Slučajan
13. Cummulative Sums (Forward) Test		0. 6652292818518775	Slučajan
Cummulative Sums (Reverse) Test		0. 5356398131057647	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**

-4	10.633797457262324	0.059144067606066	Slučajan
-3	6.4180935456831545	0.26763464996791564	Slučajan
-2	7.840085685014436	0.1652718221483846	Slučajan
-1	5.316848281642917	0.3784471382997749	Slučajan
+1	2.1936295054484494	0.8217556130163672	Slučajan
+2	5.807653700081753	0.32538728136206263	Slučajan
+3	1.238801005867559	0.9411012176288212	Slučajan
+4	4.699849147795013	0.4535972957289557	Slučajan
15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	1333	0.4869715052561189	Slučajan
-8.0	1243	0.7915525152267591	Slučajan
-7.0	1180	0.9411586959200606	Slučajan
-6.0	1160	0.8385911133491124	Slučajan
-5.0	1178	0.918470089003341	Slučajan
-4.0	1170	0.8587475672920791	Slučajan
-3.0	1140	0.6275068609271275	Slučajan
-2.0	1166	0.7496277766443249	Slučajan
-1.0	1202	0.85381740533346	Slučajan
+1.0	1200	0.8860489223013036	Slučajan
+2.0	1190	0.9717138091093858	Slučajan
+3.0	1163	0.7835747354357337	Slučajan
+4.0	1154	0.7628251216127726	Slučajan
+5.0	1177	0.9130556517920925	Slučajan
+6.0	1169	0.8822303065951109	Slučajan
+7.0	1116	0.661963626159362	Slučajan
+8.0	1088	0.5788811407640859	Slučajan
+9.0	1103	0.6549677153796281	Slučajan

U tablici 6.1.5. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. Iz rezultata je vidljivo da sekvenca prolazi svih 15 testova. Sekvenca20N sastoji se od 167772160 bitova. Vrijednost P svakog od 15 testova zadovoljava uvjet $P \geq 0.01$, iz tog razloga možemo zaključiti da je ulazna sekvenca generirana pomoću pseudoslučajnog generatora uistinu slučajna.

Prvih pet testiranih sekvenci imaju iste rezultate odnosno svih pet sekvenci veličine 1MB, 2MB, 5MB, 10 MB i 20MB uspješno prolaze svih 15 NIST-ovih statističkih testova. Iz ovih rezultata pouzdano sa 99.99% možemo reći da algoritam online generatora pseudoslučajnih brojeva u velikom postotku daje slučajne sljedove bitova (1 i 0) koje niti po jednom kriteriju od 15 testova nisu ovisni. Sve vrijednosti P daleko su od granice ≥ 0.01 pa se tako ovaj generator može koristiti kao izvor slučajnih brojeva. Kako bi potvrdili ispravnost implementacije testova slijedeće dvije sekvence trebale bi dati potpuno drugačije rezultate. Naime sekvence SekvencaTN i Sekvenca2TN nisu slučajne. Pošto je moguće testirati tekstualne datoteke, ove dvije sekvence ne bi trebale dati iste rezultate kao one generirane slučajnim generatorom zbog toga što se unutar sekvenci nalazi smislen tekst unutar kojega se neka slova, riječi pa čak i razmaci ponavljaju više puta. Samim time rezultat ne bi trebao biti slučajan.

Tablica 6.1.6. Rezultati testiranja SekvencaTN

Test		**P-Vrijednost**	**Zaključak**
01. Frequency Test (Monobit)		1.3160979285349911e-263	Nije slučajan
02. Frequency Test within a Block		9.30213995206962e-85	Nije slučajan
03. Run Test		0.0	Nije slučajan
04. Longest Run of Ones in a Block		0.0	Nije slučajan
05. Binary Matrix Rank Test		3.0624075689268023e-06	Nije slučajan
06. Discrete Fourier Transform (Spectral) Test		1.0316997626660355e-116	Nije slučajan
07. Non-Overlapping Template Matching Test		6.312717498179021e-49	Nije slučajan
08. Overlapping Template Matching Test		8.177780085258753e-59	Nije slučajan
09. Maurer's Universal Statistical test		-1.0	Nije slučajan
10. Linear Complexity Test		0.09228007739145573	Slučajan
11. Serial test:		0.0	Nije slučajan
		0. 0	Nije slučajan
12. Approximate Entropy Test		0.0	Nije slučajan
13. Cummulative Sums (Forward) Test		2.700529480108273e-266	Nije slučajan
Cummulative Sums (Reverse) Test		6.048115425884055e-264	Nije slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	8.414965986394558	0.13480040073573665	Slučajan

-3	0.9157333333333333	0.9690721629224632	Slučajan
-2	1.4417009602194786	0.9196978657507592	Slučajan
-1	10.777777777777777	0.0559684272356749	Slučajan
+1	5.666666666666667	0.34001609192154847	Slučajan
+2	3.0	0.6999858358786276	Slučajan
+3	1.8000000000000003	0.8760684003246599	Slučajan
+4	1.2857142857142856	0.9363947807905831	Slučajan
15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	4	0.7750075826013058	Slučajan
-8.0	5	0.8076705648908361	Slučajan
-7.0	5	0.7937160633795738	Slučajan
-6.0	3	0.6698153575994166	Slučajan
-5.0	3	0.6373518882339371	Slučajan
-4.0	3	0.5929800980174267	Slučajan
-3.0	8	0.9160510722818964	Slučajan
-2.0	9	1.0	Slučajan
-1.0	10	0.813663715766792	Slučajan
+1.0	1	0.05934643879191988	Slučajan
+2.0	3	0.4557843687133	Slučajan
+3.0	5	0.24548497933315	Slučajan
+4.0	5	0.79898451200478	Slučajan
+5.0	8	0.986548525482012	Slučajan
+6.0	6	0.75485125635252025	Slučajan
+7.0	3	0.495621254859657520	Slučajan
+8.0	3	0.5524586585120245	Slučajan
+9.0	10	0.32233564858510248	Slučajan

U tablici 6.1.6. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. SekvencaTN sastoji se od 128000 bitova. Međutim bitovi su ovisni, odnosno SekvencaTN je tekstualni dokument unutar kojega se nalazi smislen tekst u ASCII formatu. Tekst se zatim pretvara u binarnu sekvencu ali zbog toga što se unutar sekvence nalazi smislen tekst rezultati testiranja nisi niti slični kao kod prethodnih sekvenci. Čak 13 testova od ukupno 15 ova sekvenca

je pala. Samim time može zaključiti da sekvenca nije slučajna. 3 testa koja ova sekvenca prolazi su testovi kod kojih se ulazna sekvenca dijeli u blokove te se određeni bitovi dodaju na kraj i na početak bloka, što sekvencu koja u velikoj mjeri nije slučajna ipak dobiva nekakve naznake slučajnosti. Sekvenca pada neke od testova jer nije zadovoljen uvrijet za testiranje dok se neki testovi mogu i prekinuti pa se iz tog razloga pojavljuje vrijednost 0.0.

Tablica 6.1.7. Rezultati testiranja SekvencaT2N

Test		**P-Vrijednost**	**Zaključak**
01. Frequency Test (Monobit)		0.0	Nije slučajan
02. Frequency Test within a Block		0.0	Nije slučajan
03. Run Test		0.0	Nije slučajan
04. Longest Run of Ones in a Block		0.0	Nije slučajan
05. Binary Matrix Rank Test		1.634829915272572e-34	Nije slučajan
06. Discrete Fourier Transform (Spectral) Test		9.010746029533455e-31	Nije slučajan
07. Non-Overlapping Template Matching Test		4.5649665565311207e-20	Nije slučajan
08. Overlapping Template Matching Test		8.396305365352896e-08	Nije slučajan
09. Maurer's Universal Statistical test		-1.0	Nije slučajan
10. Linear Complexity Test		0.4702874697789938	Slučajan
11. Serial test:		0.0	Nije slučajan
		0. 0	Nije slučajan
12. Approximate Entropy Test		0.0	Nije slučajan
13. Cummulative Sums (Forward) Test		1.0	Slučajan
Cummulative Sums (Reverse) Test		1.0	Slučajan
14. Random Excursions Test:			
Stanje	**Chi kvadrat**	**P-Vrijednost**	**Zaključak**
-4	3.994446758295155	0.5502157615630898	Slučajan
-3	8.604	0.1259405350254576	Slučajan
-2	2.4362139917695473	0.7860692229597601	Slučajan
-1	5.833333333333333	0.32277564078959387	Slučajan
+1	5.833333333333333	0.32277564078959387	Slučajan
+2	1.9423868312757202	0.8570581264321775	Slučajan
+3	1.1368	0.9507949344676765	Slučajan
+4	0.6611134249618215	0.9850345063231176	Slučajan

15. Random Excursions Variant Test:			
Stanje	**Ukupan broj**	**P-Vrijednost**	**Zaključak**
-9.0	12	1.0	Slučajan
-8.0	18	0.7518296340458492	Slučajan
-7.0	17	0.7771237463572624	Slučajan
-6.0	8	0.8055405886674938	Slučajan
-5.0	7	0.7337007157654434	Slučajan
-4.0	6	0.6434288435636205	Slučajan
-3.0	8	0.7150006546880893	Slučajan
-2.0	13	0.9061856157549283	Slučajan
-1.0	15	0.5402913746074199	Slučajan
+1.0	9	0.5402913746074199	Slučajan
+2.0	15	0.7236736098317631	Slučajan
+3.0	20	0.4652088184521418	Slučajan
+4.0	19	0.5891544654500582	Slučajan
+5.0	17	0.7337007157654434	Slučajan
+6.0	16	0.8055405886674938	Slučajan
+7.0	10	0.9098500327472846	Slučajan
+8.0	5	0.7121781172552231	Slučajan
+9.0	5	0.7289281824361976	Slučajan

U tablici 6.1.6. prikazani su rezultati testiranja sekvence pomoću NIST-ovog seta testova. SekvencaT2N sastoji se od 15360 bitova. SekvencaT2N također se sastoji od smislenog teksta pa su rezultati dosta slični kao i kod SekvencaTN. SekvencaT2N je puna manja od SekvenceTN iz tog razloga prolazi nekoliko testova više, jer nije dovoljnu duga kako bi se utvrdila određena ponavljanja bitova unutar sekvence. Iz tablice je vidljivo da su neke vrijednosti P jednake nuli, razlog tome ne nepravilan odabir testiranje sekvence koja ne zadovoljava minimalne uvijete testiranja. Nekoliko testova i ova sekvenca prolazi, ali ne dovoljno kako bi se moglo reći da je slučajna.

Sekvence testirane pomoću NIST-ovog seta testova pokazale su da se sekvence generirane pomoću online generatora ponašaju kao slučajne i prolaze sve statističke testove, dok testirane

sekvence koje nisu slučajne pomoću rezultata zaista možemo vidjeti razlog zbog kojega takve sekvence padaju testove. U nastavku će se testirati još dva seta testova a to su ENT i Diehard.

Tablica 6.1.8. Rezultati testiranja Sekvenca1ED

Test	P-Vrijednost	Zaključak
Runs test	0.18786720	Slučajan
Minumum distance test	0.05665916	Slučajan
3D Spheres test)	0.37540126	Slučajan
STS Monobit test	0.68653867	Slučajan
STS Runs test	0.09958604	Slučajan
Birthdays test	0.96840169	Slučajan

Tablica 6.1.9. Rezultati testiranja Sekvenca2ED

Test	P-Vrijednost	Zaključak
Runs test	0.86812016	Slučajan
Minumum distance test	0.93615656	Slučajan
3D Spheres test)	0.72237057	Slučajan
STS Monobit test	0.33905294	Slučajan
STS Runs test	0.03918211	Slučajan
Birthdays test	0.31466604	Slučajan

Tablica 6.1.10. Rezultati testiranja Sekvenca3ED

Test	P-Vrijednost	Zaključak
Runs test	0.64866067	Slučajan
Minumum distance test	0.67260485	Slučajan
3D Spheres test)	0.57317711	Slučajan
STS Monobit test	0.49178906	Slučajan
STS Runs test	0.47647927	Slučajan
Birthdays test	0.50882799	Slučajan

U tablicama 6.1.8, 6.1.9 i 6.1.10 Prikazani su rezultati testiranja tri sekvence generirane pomoću online generatora pseudoslučajnih brojeva. Iz rezultata je vidljivo da sve tri sekvence prolaze

svih pet statističkih testova te da su vrijednosti $P \geq 0.01$. Iz toga možemo zaključiti da su testirane sekvence doista slučajne.

Tablica 6.1.11. Rezultati testiranja Sekvenca1ED

Test	Rezultat	Zaključak
Entropy	0.999999	-
Chi square distribution	0.97	Slučajan
Arithmetic mean	0.5006	Slučajan
Monte Carlo value for Pi	3.137209108	Slučajan
Serial correlation coefficient	0.001465	Slučajan

Tablica 6.1.12. Rezultati testiranja Sekvenca2ED

Test	Rezultat	Zaključak
Entropy	1.000000	-
Chi square distribution	0.00	Slučajan
Arithmetic mean	0.5000	Slučajan
Monte Carlo value for Pi	3.121634014	Slučajan
Serial correlation coefficient	-0.001445	Slučajan

Tablica 6.1.13. Rezultati testiranja Sekvenca3ED

Test	Rezultat	Zaključak
Entropy	1.000000	-
Chi square distribution	0.02	Slučajan
Arithmetic mean	0.4999	Slučajan
Monte Carlo value for Pi	3.148666667	Slučajan
Serial correlation coefficient	0.000924	Slučajan

U tablicama 6.1.11, 6.1.12 i 6.1.13 prikazani su rezultati testiranja tri sekvence generirane pomoću online generatora pseudoslučajnih brojeva. Iz rezultata je vidljivo da sve tri sekvence prolaze svih pet statističkih testova. Iz toga možemo zaključiti da su testirane sekvence doista slučajne.

Zaključak

Slučajni brojevi u današnje vrijeme vrlo često se koriste i primjenjuju se u raznim područjima kao što su računalne simulacije, numerička analiza, programiranje, kriptografija i zabava. Najčešće se koriste za zaštitu podataka i za potrebe kriptografije gdje su slučajni brojevi od velike važnosti jer se koriste kao ključevi potrebni za šifriranje podataka koji se potom šalju putem nesigurnog komunikacijskog kanala. Međutim brojevi sami po sebi nisu slučajni. U idealnom svijetu koristili bi se pravi slučajni brojevi, no kako svijet nije idealan, tako je i prave slučajne brojeve jako teško pronaći jer se u svakodnevnom životu pojavljuju jedinu u prirodnim pojavama i situacijama koje nikako nije moguće predvidjeti.

Zbog sve veće potrebe za slučajnim brojevima potrebno ih je dobiti na neki drugi način i to uz pomoć računala odnosno generatora. Generirani brojevi dobiveni pomoću generatora slučajnih brojeva su statistički nezavisni i nepredvidivi, ali pošto se takvi brojevi generiraju po nekome algoritmu ipak se može zaključiti da ti brojevi nisu sasvim slučajni, iz tog razloga pravilnije ih je zvati pseudoslučajnim brojevima. Pseudoslučajni brojevi su brojevi generirani pomoću nekog računalnog algoritma i na prvi pogled se čini zaista slučajnim ali to zapravo nisu. U ovome radu obrađena su četiri najčešća generatora slučajnih brojeva, a to su : LCF, LFG, ANSI X9.17 i RSA generator. Svaki generator ima svoje prednosti i mane, LCF i LFG generatori su jednostavni i brzi, ali nisu toliko sigurni, dok su RSA i ANSI X9.17 nešto sporiji ali i sigurniji.

Nakon što se pomoću nekog od generatora generira slučajan niz brojeva isti je potrebno podvrgnuti statističkim testovima kako bi se dokazala njegova slučajnost. U ovom radu opisana su tri skupa testova, ENT, Diehard i NIST, te su napravljena testiranja pseudoslučajnih sekvenci generiranih pomoću online generatora slučajnih brojeva. Zbog toga što je u današnje vrijeme sigurna komunikacija preko nesigurnog komunikacijskog kanala vrlo čest oblik komunikacije, razvoj algoritama za generiranje slučajnih brojeva i testova koji će tu slučajnost potvrditi još dugo vremena neće biti zaustavljen.

Literatura

[1] I. Urbiha, „*Generiranje niza pseudoslučajnih brojeva*“, Matematičko-fizički list, 2011.

[2] M.Huzak, *„Vjerojatnost i matematička statistika“*, Zagreb, 2006
https://aktuari.math.pmf.unizg.hr/docs/vms.pdf

[3] Nepoznati autor, „“Osnovni pojmovi teorije informacije“ ,
https://element.hr/artikli/file/1357

[4] D. E. Knuth, „*The art of computer programing – third edition*“, Addison-Wesley, Massachuestts, 1997.

[5] S.Jurić, „Generatori pseudo-slučajnih brojeva, Zagreb, 2001
https://sigurnost.zemris.fer.hr/random/2001_juric/

[6]A. Menezes, P. van Oorschot, S. Vanstone; „ *Handbook of Applied Cryptography*“, CRC Press, 1996.

[7] D. R. Stinson, „*Cryptography – theory and practice, third edition*“ , Chapman & Hall/CRC, Ontario, Canada, 2006.

[8] S.Lazović „Kriptografija“, Beograd, 2015

[9]A. Dujella, M. Maretić, „*Kriptografija*“, Elemental, Zagreb, 2007

[10] J. Walker *„ENT A Pseudorandom Number Sequence Test Program“* ,
2008.http://www.fourmilab.ch/random/

[11] Nepoznati autor, *„Dieharder test descriptions“*
https://sites.google.com/site/astudyofentropy/background-information/the-tests/dieharder-test-descriptions

[12] Grupa autora; *„A statistical test suite for random and pseudorandom number generators for cryptographic aplications“,* NIST, U.S. Department of Commerce, 2010.
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf

Sažetak

Generirani brojevi dobiveni pomoću generatora slučajnih brojeva su statistički nezavisni i nepredvidivi, ali pošto se takvi brojevi generiraju po nekome algoritmu ipak se može zaključiti da ti brojevi nisu sasvim slučajni. U idealnom svijetu koristili bi se pravi slučajni brojevi, no kako svijet nije idealan, tako je i prave slučajne brojeve jako teško pronaći. U ovome radu obrađena su četiri najčešća generatora slučajnih brojeva, a to su : LCF, LFG, ANSI X9.17 i RSA generator. Kako bi se provjeriti slučajnost generiranog niza koriste se standardizirani testovi slučajnosti kao što su ENT, Diehard i NIST koji su također opisani u radu. Pomoću ovih setova testova moguće je utvrditi dali je generirani niz slučajan to jest dali zadovoljava određene testove slučajnosti. U ovisnosti u rezultatima testova, testirane algoritme kasnije je moguće koristiti za potrebe šifriranja u kriptografiji ukoliko su testirani nizovi zaista slučajni ili ih koristiti u neke druge svrhe gdje sigurnost nije toliko bitna stavka.

Ključne riječi: generatori slučajnih brojeva (RNG), generatori pseudoslučajnih brojeva (PRNG), kriptografija, Linearni kongruencijski generator (LCG), Lagged Fibonacci generator (LFG), ANSI X9.17 generator, RSA generator, standardizirani testovi slučajnosti, ENT, Diehard, NIST.

Abstract

Generated numbers obtained by a random number generator are statistically independent and unpredictable, but since such numbers are generated by some algorithm, it can be concluded that these numbers are not quite random. In ideal world we would use real random numbers, but the world is not ideal, so real random numbers are really hard to find. In this paper, four most common random number generators are described: LCF, LFG, ANSI X9.17 and RSA generator. In order to check the randomness of generated sequence, standardized random tests such as ENT, Diehard and NIST are used, which are also described in the paper. Using these sets of tests, it is possible to determine whether the generated sequence is random or not. Depending on test results, tested algorithms can later be used for encryption purposes in cryptography, if the tested arrays are really random, or they can be used for other purposes where security is not so important.

Keywords: random number generators (RNGs), pseudorandom number generators (PRNGs), cryptography, Linear Congruent Generator (LCG), Lagged Fibonacci Generator (LFG), ANSI X9.17 Generator, RSA Generator, Standardized Random Tests, ENT, Diehard, NIST.

Životopis

Nebojša Petrović rođen je 12.siječnja 1994. godine u Somboru. Osnovnu školu upisao je 2000.godine u Kneževu. U 2008. Godini upisuje srednju školu u Belom Manastiru, smjer računalni tehničar. Tokom srednjoškolskog obrazovanja bio je na natjecanjima iz informatike, matematike i osnova elektrotehnike. U 2012. godini upisuje Stručni studij Informatike na Elektrotehničkom fakultetu u Osijeku, te isti završava 2015. godine. Odmah potom upisuje razlikovnu godinu smjer komunikacije i informatika te istu završava 2016. godine kada upisuje diplomski studij u Osijeku, smjer Mrežne tehnologije.

Potpis:

_____