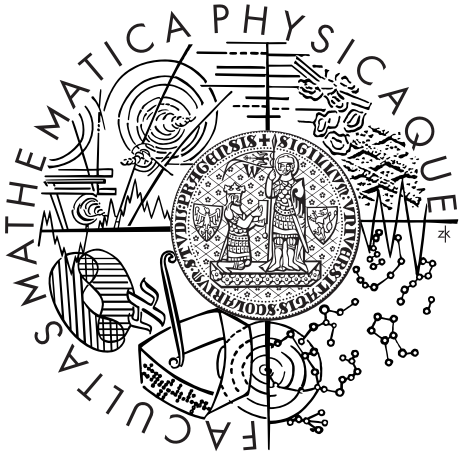
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Olha Jurečková

Testy generátorů pseudonáhodných čísel

Katedra algebry

Vedoucí diplomové práce: doc. Mgr. Pavel Příhoda, Ph.D.
Studijní program: Matematika
Studijní obor: Matematické metody informační bezpečnosti

Praha 2015

Děkuji svému vedoucímu diplomové práce doc. Mgr. Pavlu Příhodovi, Ph.D. za cenné připomínky a rady při vypracování této práce. Také bych ráda poděkovala své rodině, že mě po celou dobu studia podporovali.

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Testy generátorů pseudonáhodných čísel

Autor: Olha Jurečková

Katedra: Katedra algebry

Vedoucí diplomové práce: doc. Mgr. Pavel Příhoda, Ph.D.

Abstrakt: V předložené práci se zabýváme testy generátorů pseudonáhodných bitů. Generátory pseudonáhodných bitů jsou jedním z nejdůležitějších kryptografických nástrojů. V první části této práce uvádíme základní definice a tvrzení z teorie pravděpodobnosti a statistiky potřebné k testování náhodnosti. Dále uvedeme některé základní pojmy a fakta z kryptografie. V druhé části této práce popíšeme deset různých statistických testů a jejich modifikace. Také uvádíme výsledky testů provedených na proudové šifře Decim, Geffe generátoru a Blum Blum Shub generátoru.

Klíčová slova: PRBG, náhodnost, statistické testování

Title: Tests for generators of pseudorandom numbers

Author: Olha Jurečková

Department: Department of Algebra

Supervisor: doc. Mgr. Pavel Příhoda, Ph.D., Department of Algebra

Abstract: In this work we focus on tests for generators of pseudorandom bits. Generators of pseudorandom bits are one of the most important cryptographic tools. In the first part of this work we introduce statistical theory related for randomness testing. Then we present some basic definitions and facts from cryptography. In the second part of the work we describe ten different statistical tests and their modifications. We also present results of tests performed on Decim stream cipher, Geffe generator and Blum Blum Shub generator.

Keywords: PRBG, randomness, statistical testing

Obsah

Úvod

Náhodná čísla hrají velmi důležitou roli v mnoha různých aplikacích, např. teorii her, fyzikálních simulacích, kryptografii, atd. Náhodné posloupnosti můžeme získat na základě nějakých fyzikálních jevů, jako například šum, radioaktivní rozpad a nebo pomocí deterministických algoritmů. Generátory pseudonáhodných posloupností jsou jedním z nejdůležitějších kryptografických nástrojů. Bezpečnost mnohých kryptografických systémů závisí na použití nepředvidatelných složek jako např. keystream proudových šifer, který by se měl co nejvíce přibližovat náhodné posloupnosti bitů.

Existují různé statistické testy, které se používájí pro zjištění, jak moc se posloupnost vyprodukovaná pomocí generátoru pseudonáhodních bitů liší od náhodné posloupnosti. Poznamenejme, že procházení některými statistickými testy nemusí být dostatečným důkazem, že daný generátor pseudonáhodných bitů je dobrý.

V první části této práce uvedeme základní definice a tvrzení z teorie pravděpodobnosti a statistiky potřebné k testování náhodnosti. Dále uvedeme základní pojmy a tvrzení z kryptografie potřebné k pochopení struktury generátorů pseudonáhodných posloupností. Potom se podíváme na strukturu posuvného registru s lineární zpětnou vazbou, který je široce používaný v proudových šifrách a díky své struktuře se lehko analyzuje pomocí algebraických metod. Dále popíšeme tři generátory pseudonáhodných bitů: Blum Blum Shub generátor, Geffe generátor a Decim.

Druhá část této práce se zabývá různými druhy testů generátorů pseudonáhodných bitů. Nejprve představíme tři základní statistické testy a jejich modifikace. Ve Frekvenčním monobit testu zkoumáme, jestli počet jedniček a počet nul v testované posloupnosti je aproximačně stejný, jako v náhodné posloupnosti. V Poker testu rozdělíme vstupní posloupnost na podposloupnosti, které přiřadíme do určitých kategorií. V tomto testu používáme podobné kategorie, které se používají v karetní hře Poker. Pomocí χ^2 testu dobré shody otestujeme, jestli se naměřené počty prvků v jednotlivých kategoriích shodují s očekávanými hodnotami. V Runs testu zkoumáme celkový počet runů(souvislá posloupnost stejných bitů) různých délek v testované posloupnosti.

Dále uvedeme tři monomiální testy, které analyzují proudové šifry na základě počtu monomů algebraické normální formy(ANF) booleovské funkce. Každý výstupní bit z proudové šifry vygenerovaný pomocí tajného klíče jednoznačně vyjádříme algebraickou normální formou booleovské funkce. V Afinním konstantním testu uvažujeme několik algebraických normálních forem a testujeme, v kolika z nich je absolutní člen ANF rovný jedné. V d-monomiálním testu zkoumáme, jestli ANF booleovské funkce má očekávaný počet monomů stupně $d \geq 1$. V Monomiálním distribučním testu uvažujeme několik ANF a testujeme, v kolika z nich je každý monom obsažen.

V další kapitole se podíváme na čtyři strukturální testy, které se používají k analýze synchronních proudových šifer. V prvním testu zkoumáme korelaci mezi klíčem a odpovídajícím keystreamem. V druhém testu budeme zkoumat korelaci mezi inicializačním vektorem a odpovídajícím keystreamem. V následujícím testu uvažujeme různé inicializační vektory a podíváme se na korelaci mezi příslušnými keystreamy. V posledním testu zkoumáme difuzní vlastnosti každého bitu klíče a

inicializačního vektoru.

Nakonec uvedeme Pollardovu ρ metodu, která slouží pro faktorizaci složených čísel a představíme si způsob využití generátorů pseudonáhodných bitů pro faktorizaci čísel pomocí Pollardovy ρ metody.

1. Základy pravděpodobnosti a statistiky

Na začátku se podíváme na základní definice a věty z teorie pravděpodobnosti a statistiky. Více o základech pravděpodobnosti a statistiky najdeme v knize [1].

1.1 Základní definice

Definice 1. *Nechť Ω je neprázdná množina. Potom neprázdný systém $\mathcal{A}$ podmnožin množiny Ω se nazývá σ-algebra, pokud platí:*

1. $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$,
2. $A_n \in \mathcal{A}, n = 1, 2, \ldots \Rightarrow \bigcup_{n=1}^{\infty} A_n \in \mathcal{A}$.

Definice 2. *Nechť $\Omega = \mathbb{R}^n$. Potom σ-algebra generovaná systémem všech otevřených podmnožin v $\mathbb{R}^n$ se značí $\mathcal{B}^n$; její prvky se nazývají borelovské množiny v $\mathbb{R}^n$.*

Definice 3. *Měřitelný prostor je dvojice $(\Omega, \mathcal{A})$, kde $\mathcal{A}$ je nějaká σ-algebra podmnožin množiny Ω.*

Definice 4. *Nechť $(\Omega, \mathcal{A})$ a $(Y, \mathcal{E})$ jsou měřitelné prostory. Funkci $f : \Omega \to Y$ nazýváme měřitelná funkce, jestliže $B \in \mathcal{E} \Rightarrow f^{-1}(B) \in \mathcal{A}$.*

Definice 5. *Nechť $(\Omega, \mathcal{A})$ je měřitelný prostor. Míra μ na $(\Omega, \mathcal{A})$ je definovaná jako funkce $\mu : \mathcal{A} \to [0, \infty]$, pro kterou platí:*

1. $\mu(\emptyset) = 0$,
2. *pro každou posloupnost $(A_n \in \mathcal{A} : n \in \mathbb{N})$ po dvou disjunktních prvků z $\mathcal{A}$ platí $\mu(\bigcup_{n=1}^{\infty} A_n) = \sum_{n=1}^{\infty} \mu(A_n)$.*

Definice 6. *Prostor s mírou je trojice $(\Omega, \mathcal{A}, \mu)$, kde $(\Omega, \mathcal{A})$ je měřitelný prostor a μ je míra definována na $\mathcal{A}$.*

Definice 7. *Nechť $\Pr$ je definována jako míra na $\mathcal{A}$ s vlastností $\Pr(\Omega) = 1$. $\Pr$ nazýváme pravděpodobnostní míra. Trojici $(\Omega, \mathcal{A}, \Pr)$ říkáme pravděpodobnostní prostor.*

Definice 8. *Nechť $(\Omega, \mathcal{A}, \Pr)$ je pravděpodobnostní prostor. Měřitelná reálná funkce $X : (\Omega, \mathcal{A}) \to (\mathbb{R}, \mathcal{B})$ se nazývá náhodná veličina.*

Definice 9. *Nechť $(\Omega, \mathcal{A}, \Pr)$ je pravděpodobnostní prostor. Měřitelné zobrazení $X : (\Omega, \mathcal{A}) \to (\mathbb{R}^n, \mathcal{B}^n)$ se nazývá (n-rozměrný) náhodný vektor.*

Poznámka. Místo $\{\omega \in \Omega : X(\omega) \leq x\}$ budeme zkráceně psát $\{X \leq x\}$.

Definice 10. *Distribuční funkce F_X náhodné veličiny X je funkce $\mathbb{R} \to [0, 1]$ definována následujícím vztahem:*

$$F_X(x) = \Pr(X \leq x), \ x \in \mathbb{R}.$$

Věta 1. *Distribuční funkce F_X náhodné veličiny X má následující vlastnosti:*

1. *F_X je neklesající,*
2. $\lim_{x\to\infty} F_X(x) = 1,\ \lim_{x\to-\infty} F_X(x) = 0,$
3. *F_X je zprava spojitá.*

Definice 11. *Distribuční funkce F se nazývá diskrétní, jestliže existuje konečná nebo spočetná posloupnost reálných čísel $(x_n : n \in N_0)$, kde $N_0 \subseteq \mathbb{N}$ a odpovídající posloupnost kladných čísel $(p_n : n \in N_0)$ takových, že $\sum_{n\in N_0} p_n = 1$ a platí:*

$$F(x) = \sum_{n\in N_0; x_n \le x} p_n,\ \forall x \in \mathbb{R}.$$

Poznámka. Nechť X je náhodná veličina. Pokud X má diskrétní distribuční funkci, potom se X nazývá diskrétní náhodná veličina a říkáme, že má diskrétní rozdělení.

Definice 12. *Distribuční funkce F se nazývá spojitá, pokud existuje nezáporná měřitelná funkce $f : \mathbb{R} \to \mathbb{R}$ taková, že*

$$F(x) = \int_{-\infty}^{x} f(t)dt,\ \forall x \in \mathbb{R}.$$

Funkce f se nazývá hustota náhodné veličiny X.

Poznámka. Nechť X je náhodná veličina. Pokud X má spojitou distribuční funkci, potom se X nazývá spojitá náhodná veličina a říkáme, že má spojité rozdělení.

Definice 13. *Nechť $(\Omega, \mathcal{A}, \Pr)$ je pravděpodobnostní prostor. Nechť X je diskrétní náhodná veličina, která může nabývat jen hodnot $x_1, x_2, \ldots, x_n$. Potom střední hodnota náhodné veličiny X je:*

$$E\ X = \sum_{i=1}^{n} x_i \cdot \Pr(X = x_i).$$

Definice 14. *Nechť $(\Omega, \mathcal{A}, \Pr)$ je pravděpodobnostní prostor. Nechť X je spojitá náhodná veličina s hustotou $f(x)$. Potom střední hodnota náhodné veličiny X je:*

$$E\ X = \int_{-\infty}^{\infty} x f(x)dx.$$

Definice 15. *Nechť X je náhodná veličina definována na pravděpodobnostním prostoru $(\Omega, \mathcal{A}, \Pr)$. Rozptylem náhodné veličiny X je hodnota výrazu:*

$$var\ X = E\ (X - E\ X)^2.$$

1.2 Některá rozdělení

Dále se podíváme na rozdělení, která později budeme využívat.

Definice 16. Nula-jedničkové rozdělení. *Náhodná veličina X má nula-jedničkové rozdělení, pokud nabývá jen hodnot 0 a 1 s pravděpodobnostmi $1-p$ a p, kde $0 < p < 1$.*

Distribuční funkce F je definována následovně:

$$F(x) = \begin{cases} 0 & \text{pokud } x < 0 \\ 1-p & \text{pokud } 0 \leq x < 1 \\ 1 & \text{pokud } x \geq 1. \end{cases}$$

Střední hodnota nula-jedničkového rozdělení je $E\ X = p$ a rozptyl je $var\ X = p(1-p)$.

Definice 17. Binomické rozdělení. *Náhodná veličina X má binomické rozdělení, pokud nabývá hodnoty $k = 0, 1, \ldots, n$ s pravděpodobnostmi $\binom{n}{k}p^k(1-p)^{n-k}$, kde $n \in \mathbb{N}$ a $0 < p < 1$.*

Distribuční funkce F je definována následujícím způsobem:

$$F(x) = \begin{cases} 0 & \text{pokud } x < 0 \\ \sum_{0 \leq k \leq x} \binom{n}{k} p^k (1-p)^{n-k} & \text{pokud } 0 \leq x \leq n \\ 1 & \text{pokud } x \geq n. \end{cases}$$

Střední hodnota binomického rozdělení je $E\ X = np$ a rozptyl je $var\ X = np(1-p)$.

Definice 18. Diskrétní rovnoměrné rozdělení. *Náhodná veličina X má diskrétní rovnoměrné rozdělení na množině T, kde T je konečná a neprázdná, pokud nabývá všech hodnot z T se stejnou pravděpodobností.*

Rozdělení pravděpodobnosti náhodné veličiny X je dáno následovně:

$$\Pr(X = x) = \begin{cases} \frac{1}{|T|} & \text{pokud } x \in T \\ 0 & \text{pokud } x \notin T. \end{cases}$$

Střední hodnota diskrétního rovnoměrného rozdělení na množině T je $E\ X = \frac{1}{|T|}\sum_{x \in T} x$ a rozptyl je $var\ X = \frac{1}{|T|}\sum_{x \in T}(x - E\ X)^2$.

Definice 19. Normální rozdělení. *Náhodná veličina X má normální rozdělení $N(\mu, \sigma^2)$ s parametry μ a σ^2, kde $\mu \in \mathbb{R}$ a $\sigma^2 > 0$, pokud je hustota pravděpodobnosti ve tvaru*

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\ e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \qquad -\infty < x < \infty.$$

Střední hodnota normálního rozdělení je $E\ X = \mu$ a rozptyl je $var\ X = \sigma^2$. Pokud $X \in N(0,1)$, potom říkáme, že X má normované normální rozdělení. Distribuční funkce tohoto rozdělení se obvykle značí písmenem Φ:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}}, \qquad -\infty < x < \infty.$$

Střední hodnota normovaného normálního rozdělení je $E\ X = 0$ a rozptyl je $var\ X = 1$.

Definice 20. Chí kvadrát (χ^2-) rozdělení o n stupních volnosti. *Nechť n je celé a $n \geq 1$. Náhodná veličina X má χ^2-rozdělení o n stupních volnosti, pokud má hustotu pravděpodobnosti ve tvaru*

$$f(x) = \begin{cases} 0 & \text{pokud } x \leq 0 \\ \frac{x^{n/2-1}}{2^{n/2}\Gamma(n/2)} e^{-\frac{x}{2}} & \text{pokud } x > 0, \end{cases}$$

kde $\Gamma(n)$ je gama funkce, která je definována následovně:

$$\Gamma(n) = \int_0^\infty x^{n-1} e^{-x} dx, \qquad n > 0.$$

Střední hodnota χ^2-rozdělení o n stupních volnosti je $E\ X = n$ a rozptyl je $var\ X = 2n$.

Definice 21. *Nechť F je spojitá a ryze monotónní distribuční funkce a nechť $0 < \alpha < 1$. Potom číslo x_α takové, že $F(x_\alpha) = \alpha$ se nazývá α-kvantil příslušného rozdělení.*

Věta 2. Centrální limitní věta pro binomické rozdělení
Nechť pro každé $n \geq 1$ je X_n náhodná veličina s binomickým rozdělením o parametrech n, p, kde $0 < p < 1$. Potom

$$\lim_{n\to\infty} \Pr\left(\frac{X_n - np}{\sqrt{np(1-p)}} \leq x \right) = \Phi(x), \qquad x \in \mathbb{R}.$$

Poznámka. Náhodná veličina $Z_n = \frac{X_n - np}{\sqrt{np(1-p)}}$ má tedy pro $n >> 1$ přibližně rozdělení jako náhodná veličina Y s rozdělením $N(0,1)$. Binomická náhodná veličina $X_n = np + Z_n\sqrt{np(1-p)}$ pak má přibližně rozdělení jako $np + Y\sqrt{np(1-p)}$, tj. $N(np, np(1-p))$.

Poznámka. V [1, kap. 4] se uvádí, že v praxi lze binomické rozdělení aproximovat normálním rozdělením, pokud je splněna podmínka $np(1-p) \geq 9$.

1.3 Testování hypotéz

Statistická hypotéza je nějaké tvrzení o vlastnostech rozdělení pravděpodobnosti pozorované náhodné veličiny. Hypotézy, které se týkají hodnot parametrů rozdělení pravděpodobnosti náhodných veličin, nazýváme *parametrické hypotézy.* V opačném případě mluvíme o *neparametrických hypotézách.* Hypotézy, které jsou formulované tak, že rozdělení náhodné veličiny je jednoznačně určené, nazýváme *jednoduché hypotézy. Složené hypotézy* jsou hypotézy, které nespecifikují rozdělení náhodné veličiny jednoznačně.

Testování hypotéz je postup, který umožňuje posoudit, jestli experimentálně naměřená data splňují předpoklad, který jsme stanovili před testováním. Při testování hypotéz vždy porovnáváme dvě hypotézy. Jednu z nich, kterou testujeme, nazýváme ***nulová hypotéza*** H_0 a oproti ní stavíme tzv. ***alternativní hypotézu*** H_A.

Nechť $X = (X_1, \ldots, X_n)$ je náhodný výběr z určitého rozdělení $\mathcal{R}(\theta)$, kde θ je parametr, který může být i vícerozměrný. Nechť $h(\theta)$ je parametrická funkce a k je daná reálná konstanta. Dále nechť nulová hypotéza je v tomto tvaru:

$H_0 : h(\theta) = k.$

Alternativní hypotézu můžeme definovat následujícími třemi způsoby:

- Pravostranná alternativní hypotéza: $H_A : h(\theta) > k$
- Levostranná alternativní hypotéza: $H_A : h(\theta) < k$
- Oboustranná alternativní hypotéza: $H_A : h(\theta) \neq k$.

Během testování rozhodneme, jestli zamítneme danou hypotézu H_0 nebo nezamítneme. Při rozhodování se můžeme dopustit chyby. Když nulová hypotéza H_0 platí a na základě testu tuto hypotézu zamítneme, tak se dopustíme ***chyby prvního druhu***. Když nulová hypotéza H_0 neplatí a my ji na základě testu nezamítneme, tak se dopustíme ***chyby druhého druhu***. Můžou nastat čtyři situace, které jsou uvedené v následující tabulce.

Skutečnost	**Výsledek testu**	
	Nezamítáme H_0	Zamítáme H_0
Platí H_0	správné rozhodnutí	chyba prvního druhu
Neplatí H_0	chyba druhého druhu	správné rozhodnutí

Tabulka 1.1: Chyby při testování hypotézy

V případě jednoduché hypotézy říkáme pravděpodobnosti chyby prvního druhu ***hladina významnosti*** a obvykle ji označujeme α. V případě složené hypotézy je číslo α nejmenší horní hranicí pravděpodobnosti chyby prvního druhu.

Pokud $X = (X_1, \ldots, X_n)$ je náhodný výběr z určitého rozdělení a máme formulovanou nulovou a alternativní hypotézu, pak rozhodnutí o zamítnutí nebo nezamítnutí hypotézy H_0 zakládáme na realizaci určité statistiky $T = T(X_1, \ldots, X_n)$, kterou nazýváme ***testová statistika***. Testová statistika T je náhodná veličina, která je funkcí pozorování $X_1, \ldots, X_n$. Rozhodování probíhá tak, že zvolíme vhodnou množinu $W \subset \mathbb{R}$, které budeme říkat ***kritický obor***. V případě, že $T \in W$, hypotézu H_0 zamítneme. V opačném případě tuto hypotézu nezamítneme. Velikost kritického oboru volíme tak, abychom platnou hypotézu zamítli nejvýše s pravděpodobností α. Zpravidla se volí $\alpha = 0,05$ nebo $\alpha = 0,01$.

Při testování hypotéz se taky používá postup, při kterém určíme nejmenší hladinu významnosti, při které bychom ještě hypotézu H_0 zamítli. Této dosažené hladině významnosti říkáme ***p-hodnota***. Vyjadřuje nejmenší horní hranici pravděpodobnosti počítanou za platnosti nulové hypotézy H_0, že dostaneme právě naší realizaci testové statistiky T nebo realizaci ještě více odporující testované hypotéze.

Předpokládejme, že $X = (X_1, \ldots, X_n)$ je náhodný výběr z určitého rozdělení, $T = T(X_1, \ldots, X_n)$ je testová statistika a nechť t_0 je její hodnota. Pak p-hodnotu můžeme spočíst podle jedného ze tří možných vzorečků v závislosti na tvaru alternativní hypotézy:

- Pokud máme pravostrannou alternativní hypotézu, pak p-hodnotu spočteme podle následujícího vzorečku

$$\text{p-hodnota} = \Pr(T \geq t_0).$$

Tuto definici p-hodnoty použijeme v případech, kdy pozorovaná data svědčí o tom, že testová statistika by mohla nabývat větších hodnot, než jsou hodnoty odpovídající rozdělení testové statistiky za platnosti hypotézy H_0.

- Pokud máme levostrannou alternativní hypotézu, pak p-hodnotu spočteme podle následujícího vzorečku

$$\text{p-hodnota} = \Pr(T \leq t_0).$$

Tuto definici p-hodnoty použijeme v případech, kdy pozorovaná data svědčí o tom, že testová statistika by mohla nabývat menších hodnot, než jsou hodnoty odpovídající rozdělení testové statistiky za platnosti hypotézy H_0.

- Pokud máme oboustrannou alternativní hypotézu, pak p-hodnotu spočteme podle následujícího vzorečku

$$\text{p-hodnota} = 2 \cdot \min\left\{\Pr(T \leq t_0), \Pr(T \geq t_0)\right\}.$$

Tuto definici p-hodnoty použijeme v případech, kdy pozorovaná data svědčí o tom, že testová statistika by mohla nabývat buď větších nebo menších hodnot než jsou hodnoty odpovídající rozdělení testové statistiky za platnosti hypotézy H_0. V tomto případě můžeme uvedený způsob výpočtu p-hodnoty používat pouze tehdy, když hustota rozdělení příslušná nulové hypotéze je symetrická.

Výsledek testování hypotézy závisí na zvolené hladině významnosti α. Pokud máme spočtenou p-hodnotu, potom můžeme rozhodovat o zamítnutí nulové hypotézy na základě následující tabulky, která je založená na nejběžněji používaných hladinách významnosti $\alpha = 0,01$ nebo 0,05.

p-hodnota	**výsledek testu**
p-hodnota $< 0,01$	zamítáme H_0
$0,01 <$ p-hodnota $< 0,05$	nedokážeme rozhodnout
p-hodnota $> 0,05$	nezamítáme H_0

Tabulka 1.2: Výsledek testu na základě p-hodnoty

Příklad. Mějme následující posloupnost bitů (1110111110011011). Chceme otestovat, jestli daná posloupnost bitů je náhodná.

Řešení. Máme posloupnost bitů délky $n = 16$, kterou si reprezentujeme jako náhodný vektor $X = (X_1, \ldots, X_n)$, kde $X_i, i = 1, \ldots, 16$ je náhodná veličina s nula-jedničkovým rozdělením

$$X_i = \begin{cases} 1 & \text{s pravděpodobností } p \\ 0 & \text{s pravděpodobností } 1 - p \end{cases}$$

Daná posloupnost bitů obsahuje 12 jedniček a 4 nuly. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: $p = \frac{1}{2}$ (bit 1 je stejně pravděpodobný jako bit 0)

H_A: $p > \frac{1}{2}$

Alternativní hypotézu jsme si zvolili v uvedeném tvaru z toho důvodu, že naměřený počet jedniček je vyšší než osm. Naše testová statistika má tvar $T_{16} = \sum_{i=1}^{16} X_i$ a její hodnota je $t_0 = 12$. Za předpokladu nulové hypotézy má testová statistika T_{16} binomické rozdělení s parametry $n = 16$ a $p = \frac{1}{2}$. Potom p-hodnotu spočteme následovně:

$$\begin{aligned}
\text{p-hodnota} &= \Pr(T_{16} \geq t_0 | H_0) \\
&= \Pr(T_{16} = 12) + \ldots + \Pr(T_{16} = 16) \\
&= \binom{16}{12} \cdot \frac{1}{2^{12}} \cdot \left(1 - \frac{1}{2}\right)^{16-12} + \ldots + \binom{16}{16} \cdot \frac{1}{2^{16}} \cdot \left(1 - \frac{1}{2}\right)^{16-16} \\
&= \frac{1}{2^{16}} \left(\binom{16}{12} + \ldots + \binom{16}{16} \right) \\
&= 0,0384.
\end{aligned}$$

Jestliže použijeme předchozí tabulku pro rozhodnutí výsledku testu na základě p-hodnoty, docházíme k závěru, že pro danou posloupnost bitů nedokážeme rozhodnout, jestli je posloupnost náhodná. ■

1.4 χ^2 test dobré shody

Test dobré shody testuje shodu ***empirických četností*** (skutečné četnosti) $X_1, \ldots, X_n$ jevů $A_1, \ldots, A_n$ se středními hodnotami těchto četností (tzv. ***očekávané četnosti***) $mp_1, \ldots, mp_n$, kde pravděpodobnosti $p_1, \ldots, p_n$ jsou určeny z platnosti nějakého pravděpodobnostního modelu.

Nulová hypotéza říká, že pravděpodobnosti jevů $A_1, \ldots, A_n$ jsou po řadě rovny $p_1, \ldots, p_n$ a testová statistika má tvar:

$$X^2 = \sum_{i=1}^{n} \frac{(X_i - mp_i)^2}{mp_i}.$$

Náhodná veličina X^2 má přibližně χ^2-rozdělení o $n - 1$ stupních volnosti.[1] Nulovou hypotézu zamítáme na hladině významnosti α, jestliže $X^2 > \chi^2_{1-\alpha;\, n-1}$, kde hodnota $\chi^2_{1-\alpha;\, n-1}$ je kvantil χ^2-rozdělení o $n - 1$ stupních volnosti. Z toho potom můžeme odvodit, že naše statistika nemá χ^2-rozdělení a pravděpodobnosti jevů jsou různé od pravděpodobností $p_1, \ldots, p_n$.

Poznamenejme, že χ^2 test dobré shody je asymptotický, a proto ho možno doporučit jen při dostatečně velkém rozsahu výběru m. V literatuře se obvykle uvádí, že musí platiť $mp_i \geq 5$ pro každé $i = 1, \ldots, n$.

Příklad. Máme hrací kostku a chceme ověřit, jestli je kostka pravidelná. Hodíme kostkou 48krát a získáme následující četnosti hodů:

Hodnota	1	2	3	4	5	6
Četnost	10	6	14	2	4	12

[1] přesné znění věty spolu s důkazem najdeme v [2, kap. 12]

Řešení. Nejprve si definujme nulovou a alternativní hypotézu:

H_0: kostka je pravidelná

H_A: kostka není pravidelná

Jestli je kostka pravidelná, potom pravděpodobnost každé hodnoty je $\frac{1}{6}$. Tedy očekávané četnosti jednotlivých hodnot jsou stejné a rovné 8. Naměřené četnosti jsou $(X_1, X_2, X_3, X_4, X_5, X_6) = (10, 6, 14, 2, 4, 12)$. Dále aplikujeme test dobré shody a spočteme hodnotu:

$$\begin{aligned} X^2 &= \sum_{i=1}^{n} \frac{(X_i - mp_i)^2}{mp_i} = \sum_{i=1}^{6} \frac{(X_i - 8)^2}{8} \\ &= \frac{(10-8)^2}{8} + \frac{(6-8)^2}{8} + \frac{(14-8)^2}{8} \\ &+ \frac{(2-8)^2}{8} + \frac{(4-8)^2}{8} + \frac{(12-8)^2}{8} \\ &= 14. \end{aligned}$$

Pracujeme na hladině významnosti $\alpha = 0,05$ a uvažujeme $n - 1 = 5$ stupňů volnosti. Kvantil $\chi^2_{1-\alpha;\, n-1}$ najdeme v příslušné statistické tabulce a je rovný $\chi^2_{0,95;\, 5} = 11,071$. Máme $X^2 > 11,071$, proto nulovou hypotézu H_0 můžeme zamítnout na dané hladině významnosti. ■

2. Vybrané typy generátorů pseudonáhodných bitů

V této části si představíme tři generátory pseudonáhodných bitů, které pak otestujeme různými druhy testů. Vybrané generátory jsou: Blum Blum Shub generátor, Geffe generátor a Decim.

Proudovou šifru Decim jsme si vybrali z toho důvodu, že její autoři v článku [10] tvrdí, že je odolná vůči různým druhům útoků, speciálně i vůči algebraickému útoku. Geffe generátor se považuje za kryptograficky slabou šifru kvůli snadnému prolomení pomocí korelačního útoku[3, pozn. 6.51], avšak v článku [8] se uvádí, že s vhodně zvolenými parametry úspešně projde mnohými statistickými testy od NIST [9]. Blum Blum Shub generátor [7] je na rozdíl od předchozích dvou generátorů více algebraicky motivovaný, jeho kryptografická bezpečnost je založená na problému kvadratických zbytků.

Před samotným popisem generátorů pseudonáhodných bitů jsou uvedeny základní pojmy a tvrzení potřebné k jejich pochopení.

2.1 Základní definice a tvrzení

V této podkapitole jsou shrnuty potřebné pojmy z kryptografie[3], které používáme v této práci.

2.1.1 Generátor pseudonáhodných bitů

Bezpečnost mnohých kryptografických systémů závisí na použití nepředvidatelných složek jako např. keystream. V této části si zavedeme definici generátoru pseudonáhodných bitů (PRBG) [3, kap. 5], [7]. Nejprve si uvedeme neformální definici PRBG.

Definice 22. *(neformálně) Generátor pseudonáhodných bitů je deterministický algoritmus, který na vstupu dostane náhodnou binární posloupnost délky n a vyprodukuje binární posloupnost délky $l(n) > n$, která je statisticky nerozlišitelná od posloupnosti náhodných bitů. Vstup do PRBG nazýváme semínko(seed) a výstupu PRBG říkáme pseudonáhodná posloupnost bitů.*

Definice 23. *Řekneme, že funkce $\varepsilon : \mathbb{N} \to \mathbb{N}$ je zanedbatelná, pokud*

$$\forall k > 0 \;\; \exists n_p \;\; \forall n > n_p : \varepsilon(n) < \frac{1}{n^k}.$$

Definice 24. *Nechť X_n je náhodná veličina diskrétního rovnoměrného rozdělení na množině $\{0,1\}^n$ a Y_n je náhodná veličina diskrétního rovnoměrného rozdělení na $\{0,1\}^{l(n)}$, kde funkce $l : \mathbb{N} \to \mathbb{N}$ splňuje $l(n) > n$ pro všechna n. Nechť $G_n : \{0,1\}^n \to \{0,1\}^{l(n)}$ je posloupnost funkcí, jejíž spočtení má polynomiální složitost. Pak G_n je (kryptograficky bezpečný) PRBG, když pro všechny pravděpodobnostní algoritmy A běžící v polynomiálním čase platí, že*

$$|\Pr[A(G_n(X_n)) = 1] - \Pr[A(Y_n) = 1]|$$

je zanedbatelná funkce.

Pomocí PRBG můžeme také generovat pseudonáhodná čísla v dekadickém tvaru. K tomu stačí, aby PRBG nejdřív vygeneroval posloupnost pseudonáhodných bitů dané délky, která se pak převede na číslo v dekadickém tvaru.

2.1.2 Proudová šifra

Proudové šifry jsou důležitou součástí symetrických šifer. Vhodné jsou pro jejich snadnou hardwarovou implementaci a obecně se považují za rychlejší než blokové šifry. Následující definice jsou převzaty z [4].

Definice 25. *Nechť A je abeceda* [1] *o q symbolech, nechť $M = C$ je množina všech konečných řeťezců nad abecedou A, kde M se nazývá množina otevřených textů a C se nazývá množina šifrových textů. Dále nechť K je množina klíčů. Proudová šifra je trojice (G, E, D), kde*

- *G je generátor, který pro každý klíč $k \in K$ vytváří keystream $h_1, h_2, \ldots$, přičemž prvky h_i reprezentují libovolné transformace $E_{h_1}, E_{h_2}, \ldots$ nad abecedou A*
- *E je zobrazení, které ke každému klíči $k \in K$ přiřazuje transformaci zašifrování $E_k : M \to C$*
- *D je zobrazení, které ke každému klíči $k \in K$ přiřazuje transformaci dešifrování $D_k : C \to M$, která je vzájemně inverzní s E_k.*

Generátor G nejdřív z klíče k vygeneruje keystream $h_1, h_2, \ldots$ a pak se každý znak otevřeného textu zašifruje jinou transformací E_{h_i}. Zašifrování otevřeného textu $m = m_1, m_2, \ldots$ probíhá následovně: $c_i = E_{h_i}(m_i)$, $i = 1, 2, \ldots$ Dešifrování šifrového textu $c = c_1, c_2, \ldots$ probíhá podle vztahu $m_i = D_{h_i}(c_i), i = 1, 2, \ldots$, kde $D_{h_i} = E_{h_i}^{-1}$. V praxi se používá abeceda $A = \{0, 1\}$ a transformace zašifrování E_k a transformace dešifrování D_k jsou nejčastěji definovány pomocí operace XOR takto: $E_{h_i}(m_i) = m_i + h_i \mod 2$ a $D_{h_i}(c_i) = c_i + h_i \mod 2$.

Proudovou šifru nazýváme *synchronní*, pokud keystream nezávisí na otevřeném ani na šifrovém textu. U synchronních proudových šifer musí být příjemce i odesílatel synchronizováni, protože pokud vypadne nebo přibude alespoň jeden znak šifrového textu, např. z důvodu chyby na komunikačním kanálu, pak celý zbytek otevřeného textu bude chybně dešifrován.

Aby se klíč $k \in K$ nemusel příliš často měnit, pak se do inicializační fázi šifry zavedlo použití *inicializačního vektoru IV*. Inicializační vektor je přenášený v otevřené podobě a generátor G ho spolu s klíčem k použije v inicializační fázi, čímž se při stejném klíči vygeneruje vždy jiný keystream.

2.1.3 LFSR

V této části se podíváme na strukturu posuvného registru s lineární zpětnou vazbou [3, kap. 6], který se používá u mnohých proudových šifer. Na začátku si představíme lineární rekurentní posloupnosti [5], které použijeme při definování LFSR.

[1] neprázdná konečná množina

Definice 26. *Nechť $\mathbb{F}_q$ je konečné těleso a nechť $k \in \mathbb{N}$. Posloupnost $s = (s_0, s_1, \ldots)$ prvků tělesa $\mathbb{F}_q$ se nazývá lineární rekurentní posloupnost řádu k, pokud pro každé nezáporné celé číslo n platí*

$$s_{n+k} = a_{k-1}s_{n+k-1} + a_{k-2}s_{n+k-2} + \ldots + a_0 s_n + a, \text{ kde } a, a_0, \ldots, a_{k-1} \in \mathbb{F}_q.$$

Prvky $a_0, \ldots, a_{k-1} \in \mathbb{F}_q$ se nazývají koeficienty lineární rekurentní posloupnosti a prvek $a \in \mathbb{F}_q$ se nazývá absolutní člen lineární rekurentní posloupnosti. Pokud $a = 0$, potom lineární rekurentní posloupnost se nazývá homogenní, jinak se nazývá nehomogenní.

Definice 27. *Nechť $\mathbb{F}_q$ je konečné těleso a nechť $s = (s_0, s_1, \ldots)$ je posloupnost prvků tělesa $\mathbb{F}_q$. Když existuje $r \in \mathbb{N}$ a $n_0 \geq 0$ takové, že pro každé $n \geq n_0$ platí $s_{n+r} = s_n$, potom posloupnost s se nazývá ultimátně periodická a číslu r říkáme perioda posloupnosti.*

Definice 28. *Nechť $s = (s_0, s_1, \ldots)$ je ultimátně periodická posloupnost prvků tělesa $\mathbb{F}_q$ a nechť $r \in \mathbb{N}$ je její nejmenší perioda. Pak s se nazývá periodická, pokud pro všechna $n = 0, 1, \ldots$ platí $s_{n+r} = s_n$.*

Pro případ $\mathbb{F}_q = \mathbb{Z}_2$, kdy pracujeme s binárními posloupnostmi, si definujme pojem charakteristický polynom lineární rekurentní posloupnosti.

Definice 29. *Nechť $s = (s_0, s_1, \ldots)$ je homogenní lineární rekurentní posloupnost řádu k prvků telesa $\mathbb{Z}_2$, pro kterou platí:*

$$s_{n+k} = a_{k-1}s_{n+k-1} + a_{k-2}s_{n+k-2} + \ldots + a_0 s_n, \text{ pro } n = 0, 1, 2, \ldots,$$

kde $a_0, \ldots, a_{k-1} \in \mathbb{Z}_2$. Potom polynom $f(x) = 1 + a_{k-1}x + a_{k-2}x^2 + \ldots + a_0 x^k \in \mathbb{Z}_2[x]$ se nazývá charakteristický polynom lineární rekurentní posloupnosti s.

Příklad. Nechť máme následující homogenní lineární rekurentní posloupnost řádu 5 v $\mathbb{Z}_2$:

$$s_{n+5} = s_{n+4} + s_{n+3} + s_{n+1} + s_n, \text{ pro } n = 0, 1, 2, \ldots$$

Potom její charakteristický polynom je $f(x) = 1 + x + x^2 + x^4 + x^5$. ■

V následující definici budeme místo značení $\mathbb{F}_q$ konečného tělesa používat značení $\mathbb{F}_{p^m}$, kde p je prvočíslo, $m \in \mathbb{N}$ a $q = p^m$.

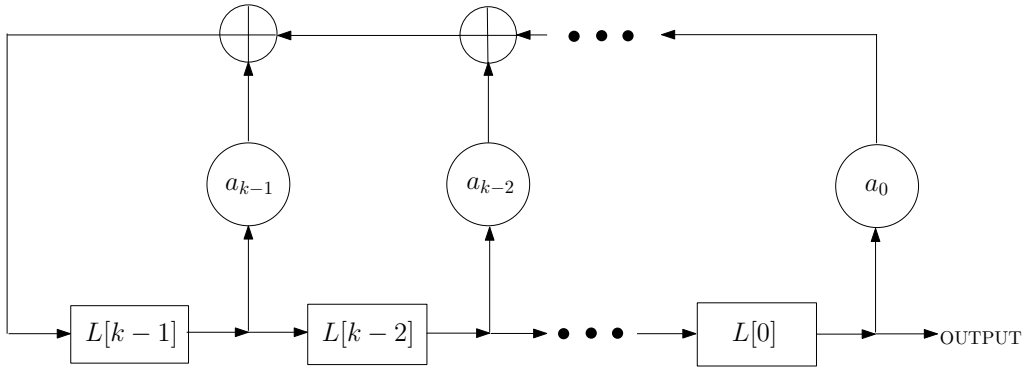
Definice 30. *Ireducibilní polynom $f(x) \in \mathbb{Z}_p[x]$ stupně m se nazývá primitivní polynom, pokud x je generátor $\mathbb{F}^*_{p^m}$, tj. multiplikativní grupy všech nenulových prvků z $\mathbb{F}_{p^m} = \mathbb{Z}_p[x]/(f(x))$.*

Definice 31. *Posuvný registr s lineární zpětnou vazbou (LFSR) délky k se skládá z k políček a mechanismu, který určuje posun registru. Každé políčko má jeden vstup a jeden výstup a obsahuje jeden bit. Jednotlivá políčka registru značíme $L[0], \ldots, L[k-1]$. Posun LFSR za jednu jednotku času definujeme následnovně:*

1. *obsah políčka $L[0]$ bude výstupem LFSR v daném čase a tvoří část výstupní posloupnosti LFSR*
2. *postupně pro $i = 1, 2, \ldots, k-1$ obsah políčka $L[i]$ přiřadíme do políčka $L[i-1]$,*

3. *nový obsah políčka $L[k-1]$ získáme pomocí zpětné vazby tak, že z předchozích obsahů políček* [2]*, které se zúčastňují zpětné vazby, se spočte jich součet modulo 2(tj. operace XOR) a výsledek se uloží do políčka $L[k-1]$.*

Struktura LFSR je znázorněna na obrázku 2.1. Konstanty $a_i \in \mathbb{Z}_2, i = 0, \ldots, k-1$ určují, která políčka $L[i]$ se použijí při výpočtu zpětné vazby. Když $a_i = 1$, pak se obsah políčka $L[i]$ použije během výpočtu zpětné vazby a když $a_i = 0$, pak se nepoužije.



Obrázek 2.1: Struktura LFSR.

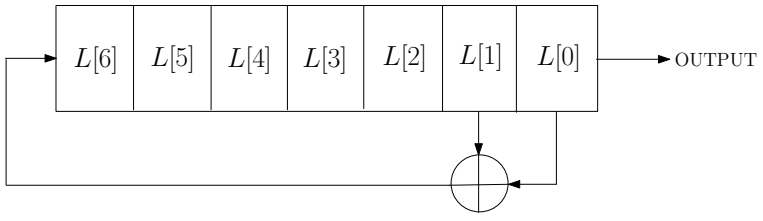
Počáteční obsah políček $L[i]$ před prvním posunem LFSR si označme jako $s_i \in \mathbb{Z}_2, i = 0, 1, \ldots, k-1$ a nazýváme ho *počáteční vnitřní stav* LFSR. V každém posunu LFSR se vyprodukuje jeden výstupní bit. Po k posunech LFSR délky k dostaneme výstupní posloupnost bitů $s_0, \ldots, s_{k-1}$. Následující výstupní bity $s_k, s_{k+1}, \ldots$ jsou dány pomocí lineární rekurentní posloupnosti řádu k:

$$s_{n+k} = a_{k-1}s_{n+k-1} + a_{k-2}s_{n+k-2} + \ldots + a_0 s_n \in \mathbb{Z}_2, \text{ pro } n = 0, 1, \ldots,$$

jejíž charakteristický polynom je $f(x) = 1 + a_{k-1}x + a_{k-2}x^2 + \ldots + a_0 x^k \in \mathbb{Z}_2[x]$. Každý LFSR je jednoznačně určen dvojicí $(k, f(x))$, kde k je délka LFSR a $f(x)$ je charakteristický polynom příslušné lineární rekurentní posloupnosti, která popisuje výstupní posloupnost daného LFSR.

Definice 32. *LFSR $(k, f(x))$ nazýváme nesingulární, pokud stupeň jeho charakteristického polynomu $f(x)$ je k.*

Příklad. Mějme LFSR délky 7 ilustrovaný na obrázku 2.2.



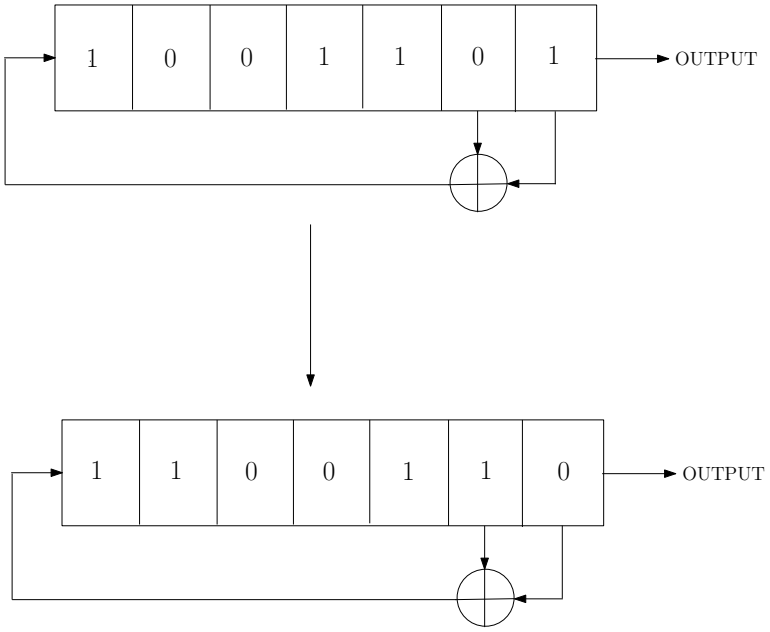
Obrázek 2.2: Příklad LFSR.

Z obrázku vidíme, že jenom políčka $L[0]$ a $L[1]$ se zúčastní zpětné vazby. Výstupní posloupnost bitů LFSR je pak dána následující lineární rekurentní posloupností délky 7 :

$$s_{n+7} = s_{n+1} + s_n, n = 0, 1, 2, \ldots$$

[2]tj. před vykonaním bodu 2.

s charakteristickým polynomem $f(x) = 1 + x^6 + x^7$. Obrázek 2.3 ilustruje pro konkrétní hodnoty bitů jeden posun LFSR, během kterého se vyprodukoval výstupní bit s hodnotou 1. ■



Obrázek 2.3: Posun LFSR.

Nakonec si uvedeme několik důležitých vlastností LFSR [3, kap. 6]:

- Počet všech nenulových počátečních vnitřních stavů LFSR délky k je $2^k - 1$.
- Každá výstupní posloupnost bitů z LFSR $(k, f(x))$ je periodická právě tehdy, když $f(x)$ má stupeň k. Když stupeň $f(x)$ je menší než k, pak výstupní posloupnost bitů z LFSR je ultimátně periodická.
- Nechť LFSR $(k, f(x))$ je nesingulární. Potom platí:
  1. pokud $f(x)$ je irreducibilní polynom nad $\mathbb{Z}_2$, potom každý nenulový počáteční vnitřní stav LFSR produkuje výstupní posloupnost s periodou rovnou nejmenšímu kladnému celému číslu n, pro které platí, že $f(x)|1 + x^n$ v $\mathbb{Z}_2[x]$.
  2. pokud $f(x) \in \mathbb{Z}_2[x]$ je primitivní polynom, potom každý nenulový počáteční vnitřní stav LFSR produkuje výstupní posloupnost maximální možné periody $2^k - 1$.

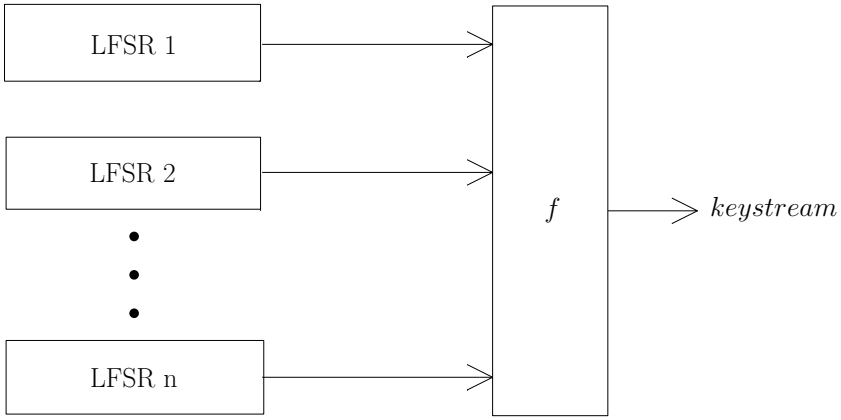
LFSR je široce používaný v proudových šifrách a díky své struktuře se lehko analyzuje pomocí algebraických metod. Výhodou LFSR je možnost produkovat posloupnosti bitů s vysokou periodou a s dobrými statistickými vlastnostmi. Další výhodou LFSR je také jeho jednoduchá hardwarová implementace.

Nelineární kombinace LFSR

Při psaní této části jsme se inspirovali práci [3, kap. 6]. Kvůli linearitě LFSR se do generátoru bitů, který používá jeden nebo více LFSR, přidává nelineární

prvek, který zvyšuje jeho kryptografickou bezpečnost. V této kapitole se stručně podíváme na konstrukci nelineární kombinace LFSR.

Uvažujme několik LFSR a zvolme si nějakou nelineární booleovskou funkci f. V každém kroku generátoru dostane funkce f na vstupu výstupní bity jednotlivých LFSR a vyprodukuje jeden bit keystreamu jako na obrázku 2.4.



Obrázek 2.4: Nelineární kombinace n LFSR.

Nelineární funkci f nazýváme *kombinující funkce*. Použitá booleovská funkce f musí splňovat kryptografické kriteria [6] kladené na odolnost vůčí základním kryptografickým útokům. Příkladem generátoru používajícího nelineární kombinaci LFSR je Geffe generátor, který je představen v podkapitole 2.2.2.

2.2 Popis vybraných generátorů

2.2.1 Blum Blum Shub generátor

V této části se podíváme na jednoduchý generátor pseudonáhodných bitů Blum Blum Shub(BBS) [7], který se považuje za kryptograficky bezpečný [3, kap. 5]. Název generátoru je odvozen od jmen autorů Lenore Blum, Manuel Blum a Michael Shub, kteří ho publikovali v roce 1968. Blum Blum Shub generátor je založen na vlastnostech Blumova celého čísla. Před samotným představením BBS si uvedeme potřebné definice a tvrzení z teorie čísel[3, kap. 2].

Definice 33. *Nechť $n \geq 2$ je kladné celé číslo a nechť $a \in \mathbb{Z}_n^*$. Řekneme, že a je kvadratickým zbytkem (reziduum) modulo n, jestliže existuje $b \in \mathbb{Z}_n^*$ takové, že $a \equiv b^2 \pmod{n}$. Když žádné takové b neexistuje, číslo a se nazývá kvadratický nezbytek. Množinu všech kvadratických zbytků modulo n značíme QR_n a množinu všech kvadratických nezbytků modulo n značíme $\overline{QR}_n$.*

Tvrzení 1. *Nechť $n = pq$, kde p a q jsou různá lichá prvočísla. Potom $a \in \mathbb{Z}_n^*$ je kvadratický zbytek modulo n právě tehdy, když $a \in QR_p$ a $a \in QR_q$.*

Definice 34. *Nechť a je kvadratický zbytek modulo n. Pak číslo $b \in \mathbb{Z}_n^*$ se nazývá druhá odmocnina a modulo n, když platí $b^2 \equiv a \bmod n$.*

Definice 35. *Nechť $a \in \mathbb{Z}$ a p je liché prvočíslo. Potom Legendreův symbol $\left(\frac{a}{p}\right)$ se definuje takto:*

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{pokud } a \in QR_p \\ -1 & \text{pokud } a \in \overline{QR}_p \\ 0 & \text{pokud } p \text{ dělí } a. \end{cases}$$

Definice 36. *Nechť $n \geq 3$ je kladné celé číslo a nechť $n = {p_1}^{e_1}{p_2}^{e_2} \dots {p_k}^{e_k}$ je jeho prvočíselný rozklad. Potom Jacobiho symbol pro $a \in \mathbb{Z}$ definujeme následovně:*

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \dots \left(\frac{a}{p_k}\right)^{e_k}.$$

Poznámka. Jacobiho symbol je zobecněním Legendreova symbolu. Pokud je n liché prvočíslo, pak se Jacobiho symbol rovná Legendreovu symbolu.

Definice 37. *Nechť $n \in \mathbb{N}$ je složené číslo ve tvaru $n = pq$, kde p a q jsou dvě různá prvočísla taková, že $p \equiv 3 \mod 4$ a také $q \equiv 3 \mod 4$. Takové číslo n se nazývá Blumovo celé číslo.*

Tvrzení 2. *Nechť $n = pq$ je Blumovo celé číslo a nechť $a \in QR_n$. Potom a má přesně čtyři druhé odmocniny modulo n a právě jedna z nich patří do QR_n.*

Tvrzení 3. *Nechť $n = pq$ je Blumovo celé číslo. Potom funkce $f : QR_n \to QR_n$ definovaná jako $f(x) = x^2 \mod n$ je permutace. Inverzní funkce k funkci f je $f^{-1}(x) = x^{\frac{(p-1)(q-1)+4}{8}} \mod n$.*

Blum Blum Shub generátor funguje následovně. V inicializační fázi si zvolíme dvě různá náhodná prvočísla p a q, která jsou kongruentní 3 modulo 4. Dále vypočteme jejich součin, tj. Blumovo celé číslo $n = pq$ a zvolíme počáteční hodnotu s z intervalu $[1, n-1]$, pro kterou platí $\text{NSD}(s, n) = 1$. Tím skončí inicializační fáze generátoru. Dále spočteme hodnotu $x_0 = s^2 \mod n$, která je kvadratickým zbytkem modulo n. Dále nastává fáze generování, v které BBS generuje bity pomocí rekurzivního výrazu ${x_{i+1}} = {x_i}^2 \mod n$. Výstupem BBS generátoru je pseudonáhodná posloupnost bitů $z_1, z_2, \dots, z_l$ pro které platí, že z_i se rovná nejméně významnému bitu hodnoty x_i v binárním zápisu. BBS generátor je shrnut v Algoritmu 1.

Algoritmus 1 : Algoritmus Blum Blum Shub

VSTUP: $l-$délka generované posloupnosti
VÝSTUP: $z_1, z_2, \dots, z_l \in \mathbb{Z}_2$

1: zvol různá náhodná prvočísla p a q, které jsou kongruentní 3 modulo 4 a spočti $n = pq$
2: zvol počáteční hodnotu s z intervalu $[1, n-1]$ takovou, že $\text{NSD}(s, n) = 1$ a spočti $x_0 = s^2 \mod n$.
3: **for** $i = 1$ **to** l **do**
4: $\quad x_i = x_{i-1}^2 \mod n$
5: $\quad z_i =$ nejméně významný bit x_i
6: **end for**
7: **return** výstupní posloupnost $z_1, z_2, \dots, z_l$

Bezpečnost algoritmu Blum Blum Shub je založena na problému nalezení kvadratických zbytků [3, kap. 3.4]. Před představením problému kvadratických zbytků si uveďme definici množiny $\mathbb{J}_n$.

Definice 38. *Nechť $n \geq 3$ je liché celé číslo. Potom množinu všech $a \in \mathbb{Z}_n^*$, pro které platí, že Jacobiho symbol $\left(\frac{a}{n}\right) = 1$, označíme $\mathbb{J}_n$.*

Poznámka. Jacobiho symbol $\left(\frac{a}{n}\right)$ na rozdíl od Legendreova symbolu neobsahuje informaci o tom, jestli a je nebo není kvadratickým zbytkem modulo n. Platí, že když $a \in QR_n$, potom $\left(\frac{a}{n}\right) = 1$. Avšak $\left(\frac{a}{n}\right) = 1$ neimplikuje, že $a \in QR_n$.

Problém kvadratických zbytků: nechť n je kladné liché složené číslo a nechť $a \in \mathbb{J}_n$, potom rozhodněte, jestli a je kvadratický zbytek modulo n.

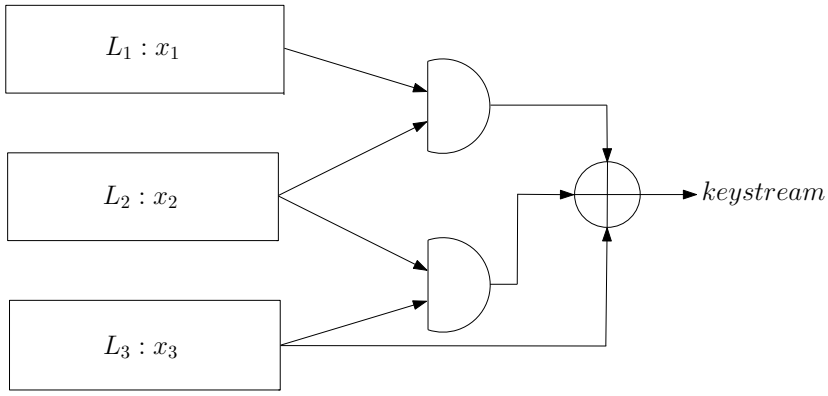
Poznámka. Problém kvadratických zbytků pro $a \in \mathbb{Z}_n^*$, kde n je prvočíslo, je snadné vyřešit díky vztahu: $\left(\frac{a}{n}\right) = 1$ platí, právě když $a \in QR_n$. Pro výpočet Legendreova symbolu $\left(\frac{a}{n}\right)$ existuje efektivní algoritmus [3, kap. 2, Algoritmus 2.149]. Když předpokládáme, že n je ve tvaru $n = pq$, kde p a q jsou dvě různá lichá prvočísla, pak podle Tvrzení 2.1 platí: když $a \in \mathbb{J}_n$, pak $a \in QR_n$ právě tehdy, když $\left(\frac{a}{p}\right) = 1$. Proto když známe faktorizaci n, pak problém kvadratických zbytků můžeme snadno vyřešit spočtením Legendreova symbolu $\left(\frac{a}{p}\right)$.

2.2.2 Geffe generátor

Geffe generátor [8] sestává ze tří LFSR, označme si je L_1, L_2 a L_3 a jedné nelineární booleovské funkce f. Délky jednotlivých LFSR jsou po dvou nesoudělné a označme si je l_1, l_2 a l_3. Výstupní bity registrů L_1, L_2 a L_3 značíme po radě x_1, x_2 a x_3. Nelineární kombinující funkce f dostane na vstupu x_1, x_2 a x_3 a vygeneruje bit keystreamu následujícím způsobem:

$$f(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + x_3.$$

Princip Geffe generátoru je znázorněn na následujícím obrázku.



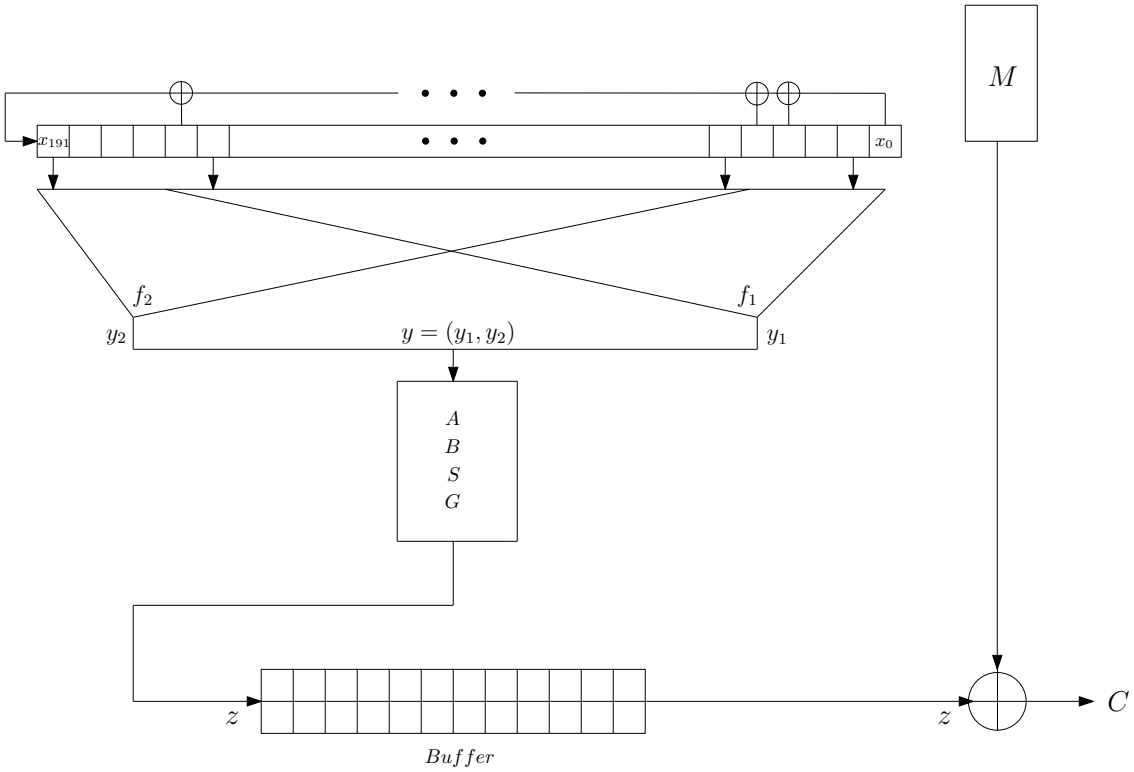
Obrázek 2.5: Geffe generátor.

Perioda keystreamu vygenerovaného Geffe generátorem je rovna $(2^{l_1} - 1) \cdot (2^{l_2} - 1) \cdot (2^{l_3} - 1)$.

2.2.3 DECIM

Decim [10] je synchronní proudová šifra, která byla navržena v rámci projektu eSTREAM [11]. Tato šifra pracuje s klíčem K délky 80 bitů a inicializačním vektorem IV délky 64 bitů, pomocí kterých šifra vygeneruje keystream z. Ten se pak po jednotlivých bitech naxoruje s otevřeným textem M, čímž vznikne šifrový text C.

Na začátku si představíme základní komponenty šifry Decim: LFSR, booleovskou funkci, mechanismy ABSG a Buffer, viz obrázek 2.6. Nakonec si uvedeme, jakým způsobem jsou uvedené komponenty šifry propojeny a jak probíhá generování keystreamu.



Obrázek 2.6: Šifra Decim.

LFSR

LFSR má délku 192 bitů, políčka registru jsou očíslovaná od 0 do 191 a značíme je $(x_0, \ldots, x_{191})$. Výstupní posloupnost z LFSR značíme jako $s = (s_t), t \geq 0$. Charakteristický polynom nad tělesem $\mathbb{Z}_2$ příslušný LFSR pro šifru Decim je

$$p(x) = x^{192}+x^{189}+x^{188}+x^{169}+x^{156}+x^{155}+x^{132}+x^{131}+x^{94}+x^{77}+x^{46}+x^{17}+x^{16}+x^5+1$$

a odpovídajíci lineární rekurentní posloupnost řádu 192 má tvar

$$s_{192+n} = s_{187+n} \oplus s_{176+n} \oplus s_{175+n} \oplus s_{146+n} \oplus s_{115+n} \oplus s_{98+n} \oplus s_{61+n}$$

$$\oplus s_{60+n} \oplus s_{37+n} \oplus s_{36+n} \oplus s_{23+n} \oplus s_{4+n} \oplus s_{3+n} \oplus s_n.$$

Booleovská funkce

Booleovská funkce f sedmi proměnných slouží jako filtr vnitřního stavu LFSR a je definovaná následujícím způsobem:

$$f(x_{i_1}, \ldots, x_{i_7}) = \sum_{1 \leq j < k \leq 7} x_{i_j} x_{i_k}.$$

Pro jeden posun LFSR se booleovská funkce f použije hned dvakrát(označená jako f_1 a f_2), aby se zvýšil počet vygenerovaných bitů. Nechť $x_{i_1}, \ldots, x_{i_{14}}$ je 14 bitů vnitřního stavu LFSR, které jsou vstupem do funkcí f_1 a f_2, kde

$$f_1(x_{i_1}, \ldots, x_{i_7}) = f(x_{i_1}, \ldots, x_{i_7}),$$

$$f_2(x_{i_8}, \ldots, x_{i_{14}}) = f(x_{i_8}, \ldots, x_{i_{14}}).$$

Potom si definujme výstupní posloupnosti z funkcí f_1 a f_2, které značíme po řadě jako $y_1 = (y_{1,t}), t \geq 0$ a $y_2 = (y_{2,t}), t \geq 0$ a které jsou definované následujícím způsobem:

$$y_{1,t} = f(s_{i_1+t}, \ldots, s_{i_7+t}),$$

$$y_{2,t} = f(s_{i_8+t}, \ldots, s_{i_{14}+t}).$$

Pro každé $t \geq 0$ se vyprodukují dva bity do posloupnosti y, která je definovaná následovně:

$$y = y_{1,0}, y_{2,0}, y_{1,1}, y_{2,1}, y_{1,2}, y_{2,2}, y_{1,3}, y_{2,3}, \ldots$$

Funkce f_1 a f_2 se používají na různých pozicích v LFSR:

- pozice pro první funkci f_1 jsou: 1, 32, 40, 101, 164, 178, 187,
- pozice pro druhou funkci f_2 jsou: 6, 8, 60, 116, 145, 181, 191.

Potom dostáváme

$$y_{1,t} = f(s_{t+1}, s_{t+32}, s_{t+40}, s_{t+101}, s_{t+164}, s_{t+178}, s_{t+187}),$$

$$y_{2,t} = f(s_{t+6}, s_{t+8}, s_{t+60}, s_{t+116}, s_{t+145}, s_{t+181}, s_{t+191}).$$

Mechanismus ABSG

Vstupní posloupnost do mechanismu ABSG [12] je $y = ((y_{1,t}, y_{2,t})), t \geq 0$ a výstupem je posloupnost z, která je vstupem do Bufferu. Mechanismus ABSG rozdělí posloupnost y na podposloupnosti tvaru $(\overline{b}, b^i, \overline{b})$, kde $i \geq 0, b \in \{0, 1\}, b^i$ označuje posloupnost $(b, b, \ldots, b)$ délky i a $\overline{b}$ znamená komplement b v $\{0, 1\}$. Pro každou podposloupnost $(\overline{b}, b^i, \overline{b})$ platí, že pokud $i = 0$, pak výstupem je bit $\overline{b}$, jinak b. V následujícím příkladě si uvedeme působení ABSG na konkrétní posloupnosti.

Příklad. Mějme 16 bitovou posloupnost $y = (1001110100001110)$. Mechanismem ABSG rozdělíme y na podposloupnosti $(\overline{b}, b^i, \overline{b})$ a vypočteme výstupní posloupnost z tímto způsobem

$$\underbrace{1001}_{0}\ \underbrace{11}_{1}\ \underbrace{010}_{1}\ \underbrace{00}_{0}\ \underbrace{01110}_{1}$$

Výstupní posloupnost ABSG je $z = 01101$. ■

Algoritmus 2 : Algoritmus ABSG

VSTUP: $(y_0, y_1, \ldots)$
VÝSTUP: z_j

$i = 0; j = 0;$
repeat:
  $e = y_i, z_j = y_{i+1}$
  $i = i + 1$
  while $y_i == \overline{e}$ **do**
    $i = i + 1$
  end while
  $i = i + 1$
  return z_j
  $j = j + 1$
end repeat

Mechanismus ABSG je shrnut v Algoritmu 2.

Mechanismus Buffer

Mechanismus ABSG je irregularní v tom smyslu, že ne po každém posunu LFSR se vygeneruje výstupní bit z_j. Abychom docílili konstantní výstup keystreamu, použijeme mechanismus Buffer, který pracuje jako fronta délky 32. Po inicializaci vnitřního stavu LFSR se Buffer postupně začne naplňovat výstupními bity z ABSG mechanismu a když se naplní, vyprodukuje 32 bitů keystreamu a pak se vynuluje pro další použití.

Inicializace vnitřního stavu LFSR

Inicializační fáze vnitřního stavu LFSR vyžaduje, aby inicializační vektor IV měl délku 80 bitů stejně jako klíč K. Protože IV má jenom 64 bitů, přidá se k němu na konci 16 nul, aby měl požadovanou délku. Vnitřní stav LFSR se vypočte následujícím způsobem:

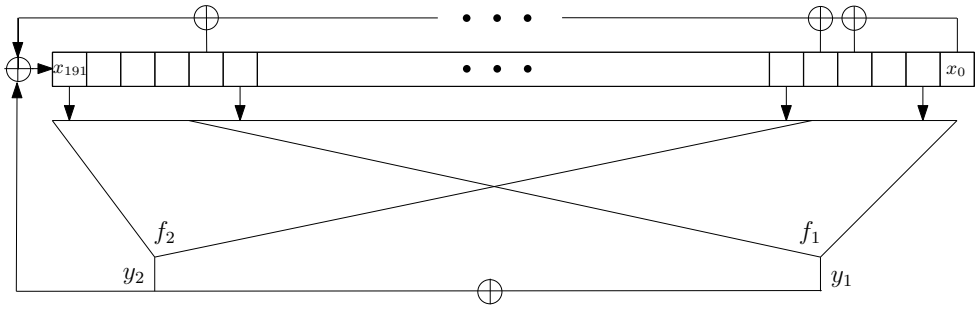
$$x_i = \begin{cases} K_i \vee IV_i & \text{pro } 0 \leq i \leq 55 \\ K_{i-56} \wedge \overline{IV}_{i-56} & \text{pro } 56 \leq i \leq 111 \\ K_{i-112} \oplus IV_{i-112} & \text{pro } 112 \leq i \leq 191 \end{cases}$$

Výpočet hodnoty x_{191} nelineární zpětné vazby

Během posunu LFSR jsou hodnoty políček $x_0, \ldots, x_{190}$ aktualizované běžným způsobem: postupně pro $i = 0, 1, \ldots, 190$ se do i- tého políčka x_i uloží hodnota políčka x_{i+1}. Avšak hodnota políčka x_{191} v čase t se spočítá následovně:

$$x_{191} = v_t \oplus y_{1,t} \oplus y_{2,t},$$

kde v_t je hodnota lineární zpětné vazby v čase $t > 0$ a $y_{1,t}, y_{2,t}$ jsou po radě výstupní hodnoty z funkcí f_1 a f_2, viz obrázek 2.7 na následující stránce.



Obrázek 2.7: Výpočet hodnoty x_{191}.

Permutace sedmi prvků

Po výpočtu hodnoty x_{191} je na vybraných sedm prvků LFSR aplikovaná jedna ze dvou permutací, které značíme π_1 a π_2. Permutace jsou aplikované na pozicích 5, 31, 59, 100, 144, 177, 186 a jsou definované následovně:

$$\pi_1 = (163)(4527) \text{ a } \pi_2 = (1473526).$$

Do ABSG vstupují v čase t dva bity $y_{1,t}$ a $y_{2,t}$. Pokud je výstupem z ABSG bit 1, potom aplikujeme permutaci π_1. Pokud ABSG v daném čase t nemá výstup nebo jeho výstupem je bit 0, pak aplikujeme permutaci π_2.

Mechanismus generování keystreamu

Nakonec si shrneme celý algoritmus generování keystreamu. Nejprve se inicializuje LFSR s použitím klíče K a inicializačního vektoru IV. Pak 192 krát opakujeme následující tři kroky:

- výpočet hodnoty x_{191} nelineární zpětné vazby,
- aplikace permutací π_1 a π_2,
- posun LFSR.

Během uvedených tří kroků se mechanismus Buffer plní bity z ABSG a když je plný, vygeneruje 32 bitů keystreamu a pak se vynuluje.

3. Základní statistické testy

V této kapitole se podíváme na základní statistické testy generátorů pseudonáhodných bitů. V prvním testu s názvem Frekvenční monobit test [3, kap. 5] zkoumáme, jestli počet jedniček a počet nul v testované posloupnosti je aproximačně stejný, jako v náhodné posloupnosti. V práci také uvádíme námi navrhnutou variantu tohoto testu. V druhém testu, který se jmenuje Poker test [13], rozdělíme vstupní posloupnost na podposloupnosti, které přiřadíme do určitých kategorií. V tomto testu používáme podobné kategorie, které se používají v karetní hře Poker. Dále pomocí χ^2 testu dobré shody otestujeme, jestli se naměřené počty prvků v jednotlivých kategoriích shodují s očekávanými hodnotami. V třetím testu, který se nazývá Runs test [15, kap. 12], zkoumáme celkový počet runů(souvislá posloupnost stejných bitů) různých délek v testované posloupnosti.

3.1 Frekvenční monobit test

Nejprve se podíváme na variantu Frekvenčního monobit testu, který používá χ^2 test dobré shody. Cílem Frekvenčního monobit testu je zjistit, jestli počet jedniček a počet nul v testované posloupnosti odpovídá příslušným počtům v náhodné posloupnosti. V dokumentu od NIST [9] se uvádí, že pokud generátor pseudonáhodných bitů selže u tohoto testu, potom s vysokou pravděpodobností selže i v dalších testech.

Předpokládejme, že testovaný generátor vyprodukoval posloupnost bitů $x = (x_1, \ldots, x_n)$. Nejprve spočteme počet jedniček v posloupnosti x a jejich počet označíme jako $n_1 = \sum_{i=1}^{n} x_i$. Počet nul bude $n_0 = n - n_1$, kde n je délka posloupnosti x. V náhodné posloupnosti bitů délky n se očekává, že počet jedniček a počet nul bude stejný a bude se rovnat $n_0 = n_1 = \frac{n}{2}$. Jinými slovy, v nulové hypotéze předpokládáme, že počet jedniček posloupnosti x je popsán náhodnou veličinou s binomickým rozdělením s parametry n a $p = \frac{1}{2}$. Dále aplikujeme χ^2 test dobré shody. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: $n_1 = \frac{n}{2}$,
H_A: $n_1 \neq \frac{n}{2}$.

Potom spočteme hodnotu testové statistiky:

$$X^2 = \frac{(n_0 - \frac{n}{2})^2}{\frac{n}{2}} + \frac{(n_1 - \frac{n}{2})^2}{\frac{n}{2}}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy rozhodneme na základě hodnoty kvantilu $\chi^2_{1-\alpha;\, f}$. Při testování hypotézy si zvolíme hladinu významnosti $\alpha = 0,05$ a uvažujeme $f = 1$ stupeň volnosti. Kvantil $\chi^2_{0,95;\, 1}$ najdeme v příslušné statistické tabulce pro χ^2- rozdělení. Hledaný kvantil se rovná $\chi^2_{0,95;\, 1} = 3,842$. Potom pokud $X^2 > \chi^2_{0,95;\, 1}$, nulovou hypotézu H_0 zamítáme na hladině významnosti 0,05. Pokud vstupní posloupnost bitů se podle testu jeví jako náhodná, pak říkáme, že generátor "prošel" testem. Uvedený Frekvenční monobit test je shrnut v Algoritmu 3 na následující straně.

Algoritmus 3 : Frekvenční monobit test

VSTUP: posloupnost bitů $x = (x_1, \ldots, x_n)$ délky n

1: spočítej počet jedniček $n_1 = \sum_{i=1}^{n} x_i$ a počet nul $n_0 = n - n_1$ posloupnosti x

2: spočítej $X^2 = \frac{(n_0 - \frac{n}{2})^2}{\frac{n}{2}} + \frac{(n_1 - \frac{n}{2})^2}{\frac{n}{2}}$

3: **if** $X^2 > \chi^2_{1-\alpha;\, f}$ **then**
    return generátor neprošel testem

4: **else**
    return generátor prošel testem

Příklad. Nechť máme nějaký generátor pseudonáhodných bitů, který vyprodukoval následující posloupnost $x = (1110100111101100)$. Chceme otestovat tento generátor pomocí výše uvedené varianty Frekvenčního monobit testu.

Řešení. Testovaná posloupnost má délku $n = 16$ a obsahuje 6 nul a 10 jedniček. Pro náhodnou posloupnost je očekávaný počet jedniček rovný očekávanému počtu nul a ten je rovný $\frac{n}{2} = 8$. Spočteme hodnotu

$$X^2 = \frac{(6-8)^2}{8} + \frac{(10-8)^2}{8} = 1.$$

Pracujeme na hladině významnosti $\alpha = 0,05$ a uvažujeme $f = 1$ stupeň volnosti, potom $\chi^2_{0,95;\,1} = 3,842$. Testovaný generátor prošel Frekvenčním monobit testem, protože $X^2 = 1 < 3,842$. ■

Dále uvedeme námi navrhnutou variantu Frekvenčního monobit testu, která je založená na Centrální limitní větě pro binomické rozdělení. Stejně jako v první variantě testu předpokládejme, že máme posloupnost bitů $x = (x_1, \ldots, x_n)$ délky n vygenerovanou generátorem, který budeme testovat. Posloupnost x budeme chápat jako náhodný výběr z nula-jedničkového rozdělení s parametrem p. Testujeme, jestli pravděpodobnost výskytu jedničky je $p_0 = \frac{1}{2}$. V praxi se binomické rozdělení aproximuje normálním, pokud $np_0(1-p_0) \geq 9$. V tomto případě stačí zvolit $n \geq 36$. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: $p = p_0$,

H_A: $p \neq p_0$.

Nejprve spočteme počet jedniček v testované posloupnosti $x = (x_1, \ldots, x_n)$ a označíme ho $n_1 = \sum_{i=1}^{n} x_i$. Dále spočteme hodnotu t testové statistiky následujícím způsobem:

$$t = \frac{\left|\frac{n_1}{n} - p_0\right|}{\sqrt{p_0(1-p_0)}}\sqrt{n}.$$

Hypotézu H_0 zamítáme na hladině významnosti α, jestli platí následující nerovnost:

$$t \geq u_{1-\frac{\alpha}{2}},$$

kde $u_{1-\frac{\alpha}{2}}$ je kvantil normovaného normálního rozdělení. Hladinu významnosti α si zvolíme $0,05$. Pak hodnota $u_{1-\frac{\alpha}{2}}$ je rovná $1,96$. Způsob, jakým zvolíme kvantil k dánému α uvedeme na konci této podkapitoly. Uvedená modifikace Frekvenčního monobit testu je shrnuta v Algoritmu 4 na následující straně.

Algoritmus 4 : Frekvenční monobit test (modifikace)

VSTUP: posloupnost bitů $x = (x_1, \ldots, x_n)$

1: spočítej počet jedniček $n_1 = \sum_{i=1}^{n} x_i$ na vstupu x

2: spočítej $t = \frac{\left|\frac{n_1}{n} - p_0\right|}{\sqrt{p_0(1-p_0)}}\sqrt{n}$

3: **if** $t \geq u_{1-\frac{\alpha}{2}}$ **then**
return generátor neprošel testem

4: **else**
return generátor prošel testem

Příklad. K danému číslu α, kde $0 < \alpha \leq 0,1$ chceme určit interval tak, aby pro náhodnou veličinu T, která má normované normální rozdělení $N(0,1)$, platilo:

1. $\Pr(T < a) = 1 - \alpha$
2. $\Pr(T > a) = 1 - \alpha$
3. $\Pr(|T| < a) = 1 - \alpha$.

Řešení. (1). Z podmínky vyplývá $1 - \alpha = \Pr(T < a) = \Phi(a)$. Odtud dostáváme, že $\Phi(a) = 1 - \alpha$ a $a = u_{1-\alpha}$.

(2). Chceme určit interval tak, aby platilo $1 - \alpha = \Pr(T > a)$. Platí $\Pr(T > a) = 1 - \Pr(T \leq a)$ a to se rovná $1 - \Phi(a)$. Odtud dostaneme, že $1 - \alpha = 1 - \Phi(a)$ a získáme $\Phi(a) = \alpha$. Potom $a = u_{\alpha}$.

(3). Z podmínky pro interval plyne $1-\alpha = \Pr(-a < T < a) = \Phi(a) - \Phi(-a) = \Phi(a) - (1 - \Phi(a)) = 2\Phi(a) - 1$. Odtud tedy dostáváme, že $2\Phi(a) - 1 = 1 - \alpha$. Úpravou získáme $\Phi(a) = 1 - \frac{\alpha}{2}$. Odkud plyne, že $a = u_{1-\frac{\alpha}{2}}$ kvantil. ■

3.2 Poker test

V této části se podíváme na Poker test a jeho modifikace. Testovanou posloupnost $x = (x_1, \ldots, x_m)$ v Poker testu rozdělíme na podposloupnosti $(x_{ni+1}, \ldots, x_{ni+n}), i = 0, 1, \ldots, \frac{m}{n} - 1$, délky n(předpokládáme, že n dělí m), které pak přiřazujeme do k kategorií.

V klasickém Poker testu používáme stejné kategorie, jaké se používají v karetní hře poker. Testovanou posloupnost prvků rozdělíme na podposloupnosti délky $n = 5$ a každou podposloupnost přiřadíme do jedné z následujících $k = 7$ kategorií:

všechny různé	*abcde*
jeden pár	*aabcd*
dva páry	*aabbc*
tři stejné	*aaabc*
full house	*aaabb*
čtyři stejné	*aaaab*
pět stejných	*aaaaa*

Předpokládejme, že prvky testované posloupnosti x jsou z množiny $\{0, 1, \ldots, 9\}$, tj. jsou cifry. Pro uvedených sedm kategorií spočteme pravděpodobnosti, že náhodná

podposloupnost cifer délky 5 patří do dané kategorie:

$$\begin{aligned}
P_1 &= \Pr(\text{všechny různé}) = 1 \cdot \frac{9}{10} \cdot \frac{8}{10} \cdot \frac{7}{10} \cdot \frac{6}{10} \cdot \binom{5}{0} = 0,3024 \\
P_2 &= \Pr(\text{jeden pár}) = 1 \cdot \frac{1}{10} \cdot \frac{9}{10} \cdot \frac{8}{10} \cdot \frac{7}{10} \cdot \binom{5}{2} = 0,5040 \\
P_3 &= \Pr(\text{dva páry}) = 1 \cdot \frac{1}{10} \cdot \frac{9}{10} \cdot \frac{1}{10} \cdot \frac{8}{10} \cdot \binom{5}{2} \cdot \binom{3}{2} \cdot \frac{1}{2} = 0,1080 \\
P_4 &= \Pr(\text{tři stejné}) = 1 \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{9}{10} \cdot \frac{8}{10} \cdot \binom{5}{3} = 0,0720 \\
P_5 &= \Pr(\text{full house}) = 1 \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{9}{10} \cdot \frac{1}{10} \cdot \binom{5}{2} = 0,0090 \\
P_6 &= \Pr(\text{čtyři stejné}) = 1 \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{9}{10} \cdot \binom{5}{4} = 0,0045 \\
P_7 &= \Pr(\text{pět stejných}) = 1 \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \binom{5}{5} = 0,0001.
\end{aligned}$$

Pro $i = 1, \ldots, 7$ spočteme očekávané počty $E_i = \frac{m}{n}.P_i$ podposloupností v jednotlivých kategoriích. Naměřené počty podposloupností testované posloupnosti cifer x si pro příslušné kategorie označme jako $X_i, i = 1, \ldots, 7$. Potom aplikujeme χ^2 test dobré shody. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: naměřené hodnoty X_i se shodují s očekávanými hodnotami E_i

H_A: naměřené hodnoty X_i se neshodují s očekávanými hodnotami E_i.

Dále spočteme hodnotu testové statistiky:

$$X^2 = \sum_{i=1}^{7} \frac{(X_i - E_i)^2}{E_i}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy H_0 rozhodneme na základě hodnoty kvantilu $\chi^2_{1-\alpha;\, f}$. Při testování hypotézy si zvolíme hladinu významnosti $\alpha = 0,05$ a uvažujeme $f = 6$ stupňů volnosti. Kvantil $\chi^2_{1-\alpha;\, f}$ je pak rovný $\chi^2_{0,95;\, 6} = 12,593$. Nulovou hypotézu zamítáme na hladině významnosti 0,05, pokud $X^2 > \chi^2_{0,95;\, 6}$.

Jednou z možností, jak modifikovat klasický Poker test, je sjednocení některých jeho kategorií. Sjednotíme kategorie **dva páry** a **tři stejné** a dostaneme kategorii **tři různé**. Dále sjednotíme kategorie **full house** a **čtyři stejné** a dostaneme kategorii **dvě různé**. Nakonec dostaneme pět následujících kategorií: **pět různých, čtyři různé, tři různé, dvě různé a jeden různý.**

V praxi se Poker test obvykle používá na posloupnosti bitů, které jsou typicky reprezentovány pomocí 32 bitových nebo 64 bitových proměnných. Protože 32 ani 64 nejsou násobkem pěti, vznikla další modifikace Poker testu, kterou si teď uvedeme. Protože v článku [13] nebyl jednoznačně popsán postup rozdělení testované posloupnosti bitů na jednotlivé podposloupnosti, uvádíme vlastní implementaci tohoto testu.

V naší implementaci této modifikace Poker testu jsme použili následující kódování: $a = 00, b = 01, c = 10$ a $d = 11$. Nejdřív rozdělíme vstupní posloupnost bitů na podposloupnosti délky dva a aplikujeme uvedené kódování. Potom nově

vzniklou posloupnost prvků z množiny $\{a, b, c, d\}$ rozdělíme na podposloupnosti délky 4 a každou přiřadíme do jedné z následujících kategorií:

všechny různé	*abcd*
jeden pár	*aabc*
dva páry	*aabb*
tři stejné	*aaab*
čtyři stejné	*aaaa*

Dále analogicky jako v klasickém Poker testu spočteme pravděpodobnosti pro uvedené kategorie:

$$\begin{aligned}
P_1 &= \Pr(\text{všechny různé}) = 1 \cdot \frac{3}{4} \cdot \frac{2}{4} \cdot \frac{1}{4} \cdot \binom{4}{0} = 0,09375 \\
P_2 &= \Pr(\text{jeden pár}) = 1 \cdot \frac{1}{4} \cdot \frac{3}{4} \cdot \frac{2}{4} \cdot \binom{4}{2} = 0,5625 \\
P_3 &= \Pr(\text{dva páry}) = 1 \cdot \frac{1}{4} \cdot \frac{3}{4} \cdot \frac{1}{4} \cdot \binom{4}{2} \cdot \binom{2}{2} \cdot \frac{1}{2} = 0,140625 \\
P_4 &= \Pr(\text{tři stejné}) = 1 \cdot \frac{1}{4} \cdot \frac{1}{4} \cdot \frac{3}{4} \cdot \binom{4}{3} = 0,1875 \\
P_5 &= \Pr(\text{čtyři stejné}) = 1 \cdot \frac{1}{4} \cdot \frac{1}{4} \cdot \frac{1}{4} \cdot \binom{4}{4} = 0,015625
\end{aligned}$$

Stejně jako v klasickém Poker testu spočteme na základě pravděpodobností P_i očekávané počty E_i podposloupností v jednotlivých pěti kategoriích. Předpokládejme, že jsme získali naměřené počty X_i podposloupností testované posloupnosti pro jednotlivé kategorie. Potom aplikujeme χ^2 test dobré shody. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: naměřené hodnoty X_i se shodují s očekávanými hodnotami E_i

H_A: naměřené hodnoty X_i se neshodují s očekávanými hodnotami E_i.

Dále spočteme hodnotu testové statistiky:

$$X^2 = \sum_{i=1}^{5} \frac{(X_i - E_i)^2}{E_i}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy H_0 rozhodneme na základě hodnoty kvantilu $\chi^2_{1-\alpha;\, f}$. Stejně jako v klasickém Poker testu pracujeme na hladině významnosti $\alpha = 0,05$, ale v tomto případě uvažujeme $f = 4$ stupně volnosti. Kvantil $\chi^2_{1-\alpha;\, f}$ je rovný $\chi^2_{0,95;\, 4} = 9,488$. Pokud $X^2 > \chi^2_{0,95;\, 4}$, potom zamítneme nulovou hypotézu H_0 na hladině významnosti 0,05. Modifikovaný Poker test shrneme v Algoritmu 5 na následující stránce.

Modifikací testu z Algoritmu 5 můžeme provést sjednocením kategorií **dva páry** a **tři stejné** a dostaneme následující čtyři kategorie: **čtyři různé, tři různé, dvě různé a jeden různý.** Pro výpočet pravděpodobností pro jednotlivé kategorie využijeme Stirlingovo číslo druhého řádu [14, kap. 7].

Definice 39. *Nechť $n, k \in \mathbb{N}$ a $n \geq k$. Stirlingovo číslo druhého řádu $S(n, k)$ označuje počet rozkladů n prvkové množiny na k neprázdných po dvou disjunktních podmnožin.*

Algoritmus 5 : Modifikovaný Poker test

VSTUP: posloupnost bitů $x = (x_1, \ldots, x_m)$ délky m

1: inicializuj počet prvků každé kategorie na nulu: $X_i = 0$, $i = 1, \ldots, 5$
2: spočítej naměřené hodnoty $X_1, \ldots, X_5$
3: spočítej očekávané hodnoty $E_1, \ldots, E_5$
4: spočítej $X^2 = \sum_{i=1}^{5} \frac{(X_i - E_i)^2}{E_i}$
5: **if** $X^2 > \chi^2_{1-\alpha;\, f}$ **then**
return generátor neprošel testem
6: **else**
return generátor prošel testem

Stirlingovo číslo druhého řádu lze spočítat pomocí následujícího vztahu:

$$S(n,k) = \frac{1}{k!} \sum_{j=0}^{k-1} (-1)^j \binom{k}{j} (k-j)^n.$$

Příklad. Najděte všechny rozklady čtyřprvkové množiny $\{1,2,3,4\}$.
Řešení. Množinu $\{1,2,3,4\}$ můžeme rozložit na:

- 1 podmnožinu jen jedním způsobem: $\{\{1,2,3,4\}\}$.
  Proto $S(4,1) = 1$.
- 2 podmnožiny 7 způsoby: $\{\{1,2\},\{3,4\}\}, \{\{1,3\},\{2,4\}\}, \{\{1,4\},\{2,3\}\}, \{\{1,2,3\},\{4\}\}, \{\{1,3,4\},\{2\}\}, \{\{1,2,4\},\{3\}\}, \{\{2,3,4\},\{1\}\}$.
  Proto $S(4,2) = 7$.
- 3 podmnožiny 6 způsoby: $\{\{1,2\},\{3\},\{4\}\}, \{\{1,3\},\{2\},\{4\}\}, \{\{1,4\},\{2\},\{3\}\}, \{\{2,3\},\{1\},\{4\}\}, \{\{2,4\},\{1\},\{3\}\}, \{\{3,4\},\{1\},\{2\}\}$.
  Proto $S(4,3) = 6$.
- 4 podmnožiny jen jedním způsobem: $\{\{1\},\{2\},\{3\},\{4\}\}$.
  Proto $S(4,4) = 1$. ■

Pomocí Stirlingova čísla druhého řádu spočteme pravděpodobnosti pro uvedené čtyři kategorie: čtyři různé, tři různé, dvě různé a jeden různý. Pro výpočet hodnot pravděpodobností použijeme následující vzorec z článku [13, kap. 2.2, vztah (1)]:

$$\Pr(k \text{ různých}) = \frac{d(d-1)\ldots(d-k+1)}{d^n} S(n,k),$$

kde $S(n,k)$ je Stirlingovo číslo druhého řádu a d značí počet různých prvků, které může posloupnost obsahovat. V našem případě uvažujeme posloupnost prvků z množiny $\{a,b,c,d\}$, proto $d = 4$. Dále uvažujeme podposloupnosti délky $n = 4$, potom pro $k \in \{1,2,3,4\}$ dostaneme následující pravděpodobnosti:

$$\begin{aligned}
P_1 &= \Pr(\text{čtyři různé}) = \frac{4(4-1)(4-2)(4-4+1)}{4^4} \cdot S(4,4) = 0,09375 \\
P_2 &= \Pr(\text{tři různé}) = \frac{4(4-1)(4-3+1)}{4^4} \cdot S(4,3) = 0,5625 \\
P_3 &= \Pr(\text{dvě různé}) = \frac{4(4-2+1)}{4^4} \cdot S(4,2) = 0,328125 \\
P_4 &= \Pr(\text{jeden různý}) = \frac{4}{4^4} \cdot S(4,1) = 0,015625.
\end{aligned}$$

Postup testování je analogický jako u předchozích modifikací Poker testu. Na základě pravděpodobností P_i spočteme očekávané počty E_i podposloupností v jednotlivých čtyřech kategoriích. Předpokládejme, že jsme získali naměřené počty X_i podposloupností testované posloupnosti pro jednotlivé kategorie. Potom aplikujeme χ^2 test dobré shody. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: naměřené hodnoty X_i se shodují s očekávanými hodnotami E_i

H_A: naměřené hodnoty X_i se neshodují s očekávanými hodnotami E_i.

Dále spočteme hodnotu testové statistiky:

$$X^2 = \sum_{i=1}^{4} \frac{(X_i - E_i)^2}{E_i}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy H_0 rozhodneme na základě hodnoty kvantilu $\chi^2_{1-\alpha;\, f}$. Pracujeme na hladině významnosti $\alpha = 0,05$, ale v tomto případě uvažujeme $f = 3$ stupně volnosti. Kvantil $\chi^2_{1-\alpha;\, f}$ je rovný $\chi^2_{0,95;\, 3} = 7,815$. Pokud $X^2 > \chi^2_{0,95;\, 3}$, potom zamítneme nulovou hypotézu H_0 na hladině významnosti 0,05.

3.3 Runs test

V této části se podíváme na Runs testy [9, 15]. Run délky k definujeme jako souvislou posloupnost k stejných bitů. Cílem tohoto testu je zjistit, jestli je v testované posloupnosti počet runů jedniček a počet runů nul různých délek stejný jako v náhodné posloupnosti.

Příklad. Posloupnost 11101001 začíná runem tří jedniček 111 a má celkově 5 následujících po sobě jdoucích runů: 111, 0, 1, 00, 1. ■

Runs test zjistí, jestli se jedničky a nuly v testované posloupnosti bitů mění velmi rychle, např. (10101010), nebo naopak velmi pomalu, např. (11110000). V dokumentu od NIST [9] se před použitím Runs testu doporučuje otestovat generátor pseudonáhodných bitů nejdřív pomocí Frekvenčního monobit testu, viz kap. 3.1. Pokud generátor selže u tohoto testu, tak není potřebné použít Runs test.

Uvažujme posloupnost bitů $x = (x_1, \ldots, x_n)$ délky n, která obsahuje n_1 jedniček a n_0 nul, kde $n = n_0 + n_1$. Nechť R je náhodná veličina, která značí celkový počet runů v posloupnosti x. Rozdělení náhodné veličiny R je následující:

- $R = 2k$: Pravděpodobnost, že náhodná veličina R nabývá $2k$ runů je rovna

$$\Pr(R = 2k) = \frac{2\binom{n_1-1}{k-1}\binom{n_0-1}{k-1}}{\binom{n}{n_1}}$$

- $R = 2k + 1$: Pravděpodobnost, že náhodná veličina R nabývá $2k + 1$ runů je rovna

$$\Pr(R = 2k+1) = \frac{\binom{n_1-1}{k}\binom{n_0-1}{k-1} + \binom{n_0-1}{k}\binom{n_1-1}{k-1}}{\binom{n}{n_1}}.$$

Příklad. Nechť M značí množinu všech binárních posloupností $x = (x_1, \ldots, x_n)$ délky n takových, že každá posloupnost obsahuje právě n_1 jedniček a $n_0 = n - n_1$ nul. Nechť R je náhodná veličina, která popisuje počet runů náhodně vybrané posloupnosti $x \in M$. Jaké je rozdělení náhodné veličiny R?

Řešení. Před samotným odvozením pravděpodobností $\Pr(R = r), r = 1, 2, \ldots, n$ poznamenejme, že následující odvození rozdělení náhodné veličiny R platí za předpokladu, že pro dané n_0 a n_1 je možné, aby posloupnost $x \in M$ mohla mít r runů[1].

Run, který se skládá z jedné nebo více jedniček, budeme nazývat *jedničkový run*. Run sestávající z jedné nebo více nul budeme nazývat *nulový run*. Odvození pravděpodobností $\Pr(R = r)$ rozdělíme na tři části podle počtu runů.

- (i) $\Pr(R = 1)$: Když se posloupnost $x \in M$ skládá ze samých jedniček nebo ze samých nul, potom $\Pr(R = 1) = 1$. Jinak je $\Pr(R = 1)$ rovná nule.

- (ii) $\Pr(R = 2k), k \in \mathbb{N}$: Uvažujme posloupnosti z M, které mají $2k$ runů, kde $k \in \mathbb{N}$. Každá taková posloupnost pak obsahuje právě k jedničkových runů a právě k nulových runů. Uvažujme posloupnost samých jedniček délky n_1 a napíšme si je do řady za sebou. Pak existuje mezi nimi $n_1 - 1$ míst, kam můžeme umístnit speciální znak tzv. oddělovač, který odděluje posloupnost jedniček na jednotlivé runy. Abychom z posloupnosti jedniček vytvořili k runů, musíme do ní umístnit $k - 1$ oddělovačů. Proto máme $\binom{n_1-1}{k-1}$ různých možností, jako rozdělit posloupnost n_1 jedniček do k jedničkových runů. Analogickým způsobem můžeme rozdělit posloupnost samých nul délky n_0 do k nulových runů $\binom{n_0-1}{k-1}$ různými způsoby. Potom dostáváme, že počet posloupností z M, které obsahují $2k$ runů a mají první run jedničkový, je rovný $\binom{n_1-1}{k-1}\binom{n_0-1}{k-1}$.

  Stejným způsobem můžeme odvodit, že počet posloupností z M obsahujících $2k$ runů a které mají první run nulový, je $\binom{n_0-1}{k-1}\binom{n_1-1}{k-1}$.

  Potom celkový počet posloupností z M s $2k$ runy je:

$$\binom{n_1 - 1}{k - 1}\binom{n_0 - 1}{k - 1} + \binom{n_0 - 1}{k - 1}\binom{n_1 - 1}{k - 1} = 2\binom{n_1 - 1}{k - 1}\binom{n_0 - 1}{k - 1}.$$

  Nakonec dostáváme, že

$$\Pr(R = 2k) = \frac{2\binom{n_1-1}{k-1}\binom{n_0-1}{k-1}}{\binom{n}{n_1}},$$

  kde jmenovatel je rovný velikosti množiny M.

- (iii) $\Pr(R = 2k + 1), k \in \mathbb{N}$:

  Uvažujme posloupnosti z M, které mají $2k + 1$ runů, kde $k \in \mathbb{N}$. Každá taková posloupnost potom obsahuje $k+1$ jedničkových nebo nulových runů.

  Uvažujme jenom posloupnosti z M, které obsahují $k + 1$ jedničkových runů z celkových $2k + 1$ runů. Analogickým způsobem jako v části (ii) máme $\binom{n_1-1}{k}$ různých způsobů, jak rozdělit posloupnost samých jedniček délky

[1] například když $n_0 = 2$ a $n_1 = 10$, pak posloupnost $x \in M$ může nabývat maximálně 5 runů

n_1 do $k+1$ jedničkových runů. Posloupnost samých nul délky n_0 můžeme stejným způsobem rozdělit do k nulových runů $\binom{n_0-1}{k-1}$ různými způsoby. Potom dostáváme, že počet posloupností z M obsahujících $k+1$ jedničkových runů z celkových $2k+1$ runů je $\binom{n_1-1}{k}\binom{n_0-1}{k-1}$.

Na druhé straně když uvažujeme posloupnosti z M, které obsahují $k+1$ nulových runů z celkových $2k+1$ runů, potom analogicky můžeme odvodit, že počet takových posloupností je $\binom{n_0-1}{k}\binom{n_1-1}{k-1}$.

Pak celkový počet posloupností z M obsahujících právě $2k+1$ runů je:

$$\binom{n_1-1}{k}\binom{n_0-1}{k-1}+\binom{n_0-1}{k}\binom{n_1-1}{k-1}.$$

Tedy dostáváme, že

$$\Pr(R=2k+1)=\frac{\binom{n_1-1}{k}\binom{n_0-1}{k-1}+\binom{n_0-1}{k}\binom{n_1-1}{k-1}}{\binom{n}{n_1}}. \qquad \blacksquare$$

Nulovou hypotézu H_0, že testovaná posloupnost bitů je náhodná, můžeme zamítnout v případě, když naměřený počet runů r je příliš vysoký nebo naopak příliš nízký. O zamítnutí nebo nezamítnutí nulové hypotézy H_0 rozhodneme na základě p-hodnoty, kterou spočteme podle následujícího vzorečku:

$$\text{p-hodnota} = 2 \cdot \min\left\{\Pr(R \le r), \Pr(R \ge r)\right\}.$$

Výsledek testu určíme na základě získané p-hodnoty podle tabulky (1.2) z první kapitoly této práce.

V [15, kap. 12] se uvádí, že pokud jsou n_0 a n_1 dostatečně velká a H_0 platí, potom rozdělení náhodné veličiny R můžeme aproximovat normálním rozdělením $N(\mu,\sigma^2)$, kde

$$\mu=\frac{2n_0n_1}{n_0+n_1}+1 \qquad \text{a} \qquad \sigma^2=\frac{2n_0n_1(2n_0n_1-n_0-n_1)}{(n_0+n_1)^2(n_0+n_1-1)}. \tag{3.1}$$

Potom dostáváme:

$$\begin{aligned}\Pr(R\le r) &= \Pr\left(\frac{R-\mu}{\sigma}\le\frac{r-\mu}{\sigma}\right)\\ &= \Pr\left(T\le\frac{r-\mu}{\sigma}\right)\\ &\approx \Phi\left(\frac{r-\mu}{\sigma}\right),\end{aligned}$$

kde testová statistika $T=\frac{R-\mu}{\sigma}$ má přibližně normované normální rozdělení $N(0,1)$.

Podobně dostáváme:

$$\Pr(R\ge r)\approx 1-\Phi\left(\frac{r-\mu}{\sigma}\right).$$

Potom p-hodnotu můžeme aproximovat podle následujícího vzorečku:

$$\text{p-hodnota} \approx 2 \cdot \min\left\{\Phi\left(\frac{r-\mu}{\sigma}\right), 1-\Phi\left(\frac{r-\mu}{\sigma}\right)\right\}.$$

V Algoritmu 6 je shrnuta verze Runs testu, v které používáme aproximaci rozdělení náhodné veličiny R. V prvním kroku se spočte počet jedniček a počet nul testované posloupnosti x. Potom v krocích 2. až 5. spočteme počet runů posloupnosti x. V následujících třech krocích spočteme hodnoty μ, σ a hodnotu t_0 testové statistiky T. Potom spočteme p-hodnotu, na základě které pak rozhodneme, jestli H_0 zamítneme nebo nezamítneme.

Algoritmus 6 : Runs test

VSTUP: posloupnost bitů $x = (x_1, \ldots, x_n)$ délky n

1: spočítej počet jedniček $n_1 = \sum_{i=1}^{n} x_i$ a počet nul $n_0 = n - n_1$ posloupnosti x
2: inicializuj počet runů $r = 1$
3: **for** $i = 1$ **to** $n - 1$ **do**
4: **if** $X_i \neq X_{i+1}$ **then** $r = r + 1$
5: **end for**
6: spočítej μ a σ podle (3.1)
7: spočítej $t_0 = \frac{r-\mu}{\sigma}$
8: spočítej p-hodnotu $= 2 \cdot \min\{\Phi(t_0), 1 - \Phi(t_0)\}$
9: **if** p-hodnota $< 0,01$ **then**
return generátor neprošel testem
10: **else**
return generátor prošel testem

4. Monomiální testy

Monomiální testy zkoumají, jestli algebraická normální forma (ANF) booleovské funkce obsahuje očekáváný počet monomů stupnů d. První, kdo navrhl analyzovat kryptografické systémy(symetrické šifry, hašovací funkce) na základě počtu monomů ANF byl Eric Filiol. Ve své práci [17] uvádí, že každý výstupní bit z proudové šifry vygenerovaný pomocí tajného klíče, můžeme jednoznačně vyjádřit algebraickou normální formou booleovské funkce. Pokud máme n bitů keystreamu z proudové šifry, potom můžeme tuto posloupnost bitů vyjádřit jako n booleovských funkcí $(f_t(K))_{0\leq t<n} = (f_0(K), \ldots, f_{n-1}(K))$, kde $f_i(K)$ značí i-tý bit keystreamu vyprodukovaný šifrou s tajným klíčem K. Každou booleovskou funkci f_i převedeme do ANF, kterou potom zkoumáme v Monomiálních testech.

Na začátku této kapitoly si zadefinujeme algebraickou normální formu booleovské funkce a podíváme se na její základní vlastnosti [19]. Potom uvedeme tři testy, které porovnávají naměřený počet monomů ANF s očekávaným počtem. V Afinním konstantním testu [17] uvažujeme několik algebraických normálních forem a testujeme, v kolika z nich je absolutní člen ANF rovný jedné. V d-monomiálním testu [18] zkoumáme počet monomů stupně d v ANF booleovské funkce. V posledním testu s názvem Monomiální distribuční test [18] uvažujeme několik ANF a testujeme, v kolika z nich je každý monom obsažen.

4.1 ANF booleovské funkce

Definice 40. *Algebraická normální forma (ANF) booleovské funkce* $f : \mathbb{F}_2^n \to \mathbb{F}_2$ *je funkce* $\widehat{f} : \mathbb{F}_2^n \to \mathbb{F}_2$ *taková, že*

$$\widehat{f}(x) = \sum_{a\in\mathbb{F}_2^n} f(a) \prod_{i=1}^{n} {x_i}^{a_i}.$$

Poznámka. Pro každou booleovskou funkci f existuje jednoznačně určena funkce $\widehat{f}$. ANF booleovské funkce f si můžeme napsat jako následující booleovský polynom:

$$a_0 \oplus a_1x_1 \oplus \ldots \oplus a_nx_n \oplus a_{n+1}x_1x_2 \oplus \ldots \oplus a_{2^n-1}x_1x_2\ldots x_n,$$

kde $a_i \in \mathbb{F}_2$. Prvek a_0 budeme v této práci nazývat *absolutní člen* ANF.

Tvrzení 4. *ANF je sama k sobě inverzní, tj. pokud* $g = \widehat{f}$*, potom* $\widehat{g} = f$.

Definice 41. *Částečné uspořádání vektorů z* $\mathbb{F}_2^n$ *je definované následujícím způsobem:* $x \leq y$*, pokud* $x_i \leq y_i$ *pro všechna* i.

Poznámka. Pokud použijeme částečné uspořádání vektorů, můžeme definici algebraické normální formy napsat následujícím způsobem: $\widehat{f}(x) = \sum_{a\leq x} f(a)$.

Příklad. Nechť máme booleovskou funkci $f : \mathbb{F}_2^3 \to \mathbb{F}_2$, která je daná následující pravdivostní tabulkou:

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	1
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Najdeme předpis booleovské funkce f a předpis její algebraické normální formy $\widehat{f}$.

Řešení. Podle definice algebraické normální formy dostaneme:

$$\widehat{f}(x_1, x_2, x_3) = f(0,0,0) + f(1,0,0)x_1 + f(0,1,0)x_2 + f(1,1,0)x_1x_2 +$$

$$f(0,0,1)x_3 + f(1,0,1)x_1x_3 + f(0,1,1)x_2x_3 + f(1,1,1)x_1x_2x_3.$$

Pokud dosadíme hodnoty z pravdivostní tabulky, potom získáme následující předpis algebraické normální formy:

$$\widehat{f}(x_1, x_2, x_3) = 1 + x_1 + x_1x_2 + x_3 + x_1x_2x_3. \tag{4.1}$$

Dále chceme najít předpis booleovské funkce f. Použijeme vlastnost, že ANF je sama k sobě inverzní. Podle definice ANF potom můžeme předpis booleovské funkce f vyjádřit následovně:

$$f(x_1, x_2, x_3) = \widehat{f}(0,0,0) + \widehat{f}(1,0,0)x_1 + \widehat{f}(0,1,0)x_2 + \widehat{f}(1,1,0)x_1x_2 +$$
$$\widehat{f}(0,0,1)x_3 + \widehat{f}(1,0,1)x_1x_3 + \widehat{f}(0,1,1)x_2x_3 + \widehat{f}(1,1,1)x_1x_2x_3.$$

Jednou z možností, jak získat předpis booleovské funkce f, je spočtení funkčních hodnot ANF podle (4.1).

Uvedeme si ještě jiný způsob, jak získat předpis booleovské funkce. Předpis booleovské funkce f můžeme získat i bez nutnosti spočtení funkčních hodnot ANF a to pomocí vyřešení následující soustavy lineárních rovnic nad $\mathbb{F}_2$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \widehat{f}(0,0,0) \\ \widehat{f}(1,0,0) \\ \widehat{f}(0,1,0) \\ \widehat{f}(1,1,0) \\ \widehat{f}(0,0,1) \\ \widehat{f}(1,0,1) \\ \widehat{f}(0,1,1) \\ \widehat{f}(1,1,1) \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Soustavu rovnic vyřešíme Gaussovou eliminační metodou a získáme následující řešení:

$$\widehat{f}(0,0,0) = 1, \widehat{f}(1,0,0) = 0, \widehat{f}(0,1,0) = 1, \widehat{f}(1,1,0) = 1,$$
$$\widehat{f}(0,0,1) = 0, \widehat{f}(1,0,1) = 1, \widehat{f}(0,1,1) = 0 \text{ a } \widehat{f}(1,1,1) = 1.$$

Odtud dostaneme následující předpis booleovské funkce:

$$f(x_1, x_2, x_3) = 1 + x_2 + x_1x_2 + x_1x_3 + x_1x_2x_3.$$

■

Definice 42. *Nechť $x = (x_1 \dots x_n) \in \mathbb{F}_2^n$. Hammingova váha vektoru x je definovaná následovně:*

$$w(x) = |\{i \in \{1, \dots, n\} | x_i = 1\}|.$$

Definice 43. *Nechť $x = (x_1 \dots x_n) \in \mathbb{F}_2^n$ a nechť $y = (y_1 \dots y_n) \in \mathbb{F}_2^n$. Hammingova vzdálenost mezi vektory x a y je definovaná následujícím způsobem:*

$$d(x, y) = |\{i \in \{1, \dots, n\} | x_i \neq y_i\}|.$$

Definice 44. *Monom $f(a)\prod_{i=1}^{n} {x_i}^{a_i}$ algebraické normální formy $\widehat{f}$ má stupeň k, pokud $f(a) = 1$ a $w(a) = k$, kde w je Hammingova váha.*

Definice 45. *d-zkrácená ANF booleovské funkce f je definována následovně:*

$$\widehat{f}_d(x) = \begin{cases} \widehat{f}(x) & \text{pokud } w(x) \leq d \\ 0 & \text{jinak} \end{cases}$$

Jinými slovy d-zkrácená ANF $\widehat{f}_d(x)$ vznikne z $\widehat{f}(x)$ vynecháním všech monomů stupně většího než d.

Tvrzení 5. *Algebraická normální forma náhodně zvolené booleovské funkce $f : \mathbb{F}_2^n \to \mathbb{F}_2$ má v průměru 2^{n-1} monomů. Pro každé k takové, že $0 \leq k \leq n$, je v průměru $\frac{1}{2}\binom{n}{k}$ monomů stupně k.*

Věta 3. *Nechť n_k je počet monomů stupně k algebraické normální formy náhodně zvolené booleovské funkce $f : \mathbb{F}_2^n \to \mathbb{F}_2$. Potom n_k má normální rozdělení se střední hodnotou $\frac{1}{2}\binom{n}{k}$ a rozptylem $\frac{1}{4}\binom{n}{k}$.*

Poznámka. Důkazy Tvrzení 5 a Věty 3 ihned plynou z faktu, že pro náhodně zvolenou booleovskou funkci $f : \mathbb{F}_2^n \to \mathbb{F}_2$ platí, že $\Pr[f(x_1, \dots, x_n) = 1] = \frac{1}{2}$ pro všechny $(x_1, \dots, x_n) \in \mathbb{F}_2^n$. Odtud dostaneme, že pravděpodobnost každého monomu ANF náhodně zvolené booleovské funkce je rovna $\frac{1}{2}$.

Výpočet ANF

Dále se podíváme na algoritmus výpočtu ANF booleovské funkce f délky n, kterou budeme mít reprezentovanou pomocí binárního vektoru délky 2^n.

Nechť $z : \mathbb{F}_2^n \to \mathbb{Z}$ je zobrazení, které převádí binární vektory do celých čísel, tj. $z(x) = \sum_{i=1}^{n} 2^{i-1}x_i$. Předpokládejme, že pravdivostní tabulka booleovské funkce f v n proměnných je zastoupena v binárním vektoru v délky 2^n, který je ve tvaru $v_{z(x)+1} = f(x)$ pro všechna x. Potom algebraickou normální formu vektoru v délky 2^n lze vypočítat pomocí Algoritmu 7, který používá dva pomocné vektory t a u délky 2^{n-1}.

Algoritmus 7 : Výpočet ANF

VSTUP: binární vektor v délky 2^n
VÝSTUP: ANF vektoru v

```
1: for j = 1 to n do
2:    for i = 1 to 2^{n-1} do
3:       t_i = v_{2i-1}
4:       u_i = v_{2i-1} ⊕ v_{2i}
5:    end for
6:    v = t||u
7: end for
```

4.2 Afinní konstantní test

Nechť K je tajný klíč a IV inicializační vektor testované proudové šifry. V Afinním konstantním testu zkoumáme n ANF booleovských funkcí $f_i, i = 1, \ldots, n$, které získáme následujícím způsobem: z K si vybereme nějakých m složek $k_1, \ldots, k_m$ a všechny ostatní bity K a všechny bity IV jsou považovány za konstantní. Pro všech 2^m možných vektorů $(k_1, \ldots, k_m)$ inicializujeme proudovou šifru pomocí K a IV a spočteme prvních n bitů keystreamu. Potom pro všech 2^m možných vektorů $(k_1, \ldots, k_m)$ budeme $f_i(k_1, \ldots, k_m)$ chápat jako i-tý bit keystreamu, který byl vygenerovaný pro daný vektor $(k_1, \ldots, k_m)$. Pro každou booleovskou funkci si spočteme její ANF a nechť p označuje pravděpodobnost, že absolutní člen ANF je rovný jedné. Potom si definujme nulovou hypotézu a alternativní hypotézu následovně:

$H_0 : p = 1/2$,
$H_A : p \neq 1/2$.

Nechť Y je náhodná veličina, která popisuje počet ANF n booleovských funkcí f_i, jejíž absolutní člen je rovny jedné. Potom nulovou hypotézu si v souladu s Větou 3 můžeme ekvivalentně definovat takto:

H_0 : náhodná veličina Y má normální rozdělení $N(\mu, \sigma^2)$ se střední hodnotou $\mu = n/2$ a rozptylem $\sigma^2 = n/4$.

Dále spočteme hodnotu t testové statistiky následujícím způsobem:

$$t = \frac{M - \frac{n}{2}}{\frac{\sqrt{n}}{2}},$$

kde M značí počet booleovských funkcí $f_i, i \in \{1, \ldots, n\}$, jejíž ANF má absolutní člen rovny jedné. Zvolíme si hladinu významnosti α a spočteme si hodnotu x_α takovou, že pro náhodnou veličinu X s normovaným normálním rozdělením platí:

$$\Pr[X > x_\alpha] = \Pr[X < -x_\alpha] = \frac{\alpha}{2}.$$

Způsob, jakým spočteme hodnotu x_α pro konkrétní hodnotu $\alpha = 0,05$ uvedeme na konci této podkapitoly.

Nulovou hypotézu H_0 zamítáme, jestli platí následující nerovnosti: $t > x_\alpha$ nebo $t < -x_\alpha$.

Afinní konstantní test shrneme do Algoritmu 8 na následující straně.

Algoritmus 8 : Afinní konstantní test

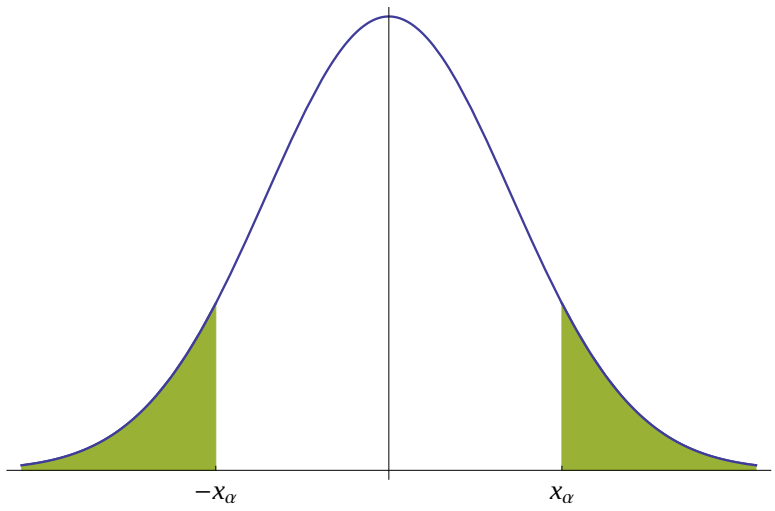
VSTUP: posloupnost bitů $x = (x_1, \ldots, x_n)$

1: vyjádři každý bit x_i pomocí booleovské funkce f_i
2: napiš každou f_i ve tvaru ANF
3: spočti M - počet ANF f_i s absolutním členem rovnym jedné
4: zvol hladinu významnosti α a spočítej x_α
5: spočítej $t = \frac{M - \frac{n}{2}}{\frac{\sqrt{n}}{2}}$
6: **if** $|t| > x_\alpha$ **then**
    return šifra neprošla testem
7: **else**
    return šifra prošla testem

Příklad. Nechť X je náhodná veličina s normovaným normálním rozdělením a nechť $\alpha = 0,05$. Spočítej hodnotu x_α takovou, že platí:

$$\Pr[X > x_\alpha] = \Pr[X < -x_\alpha] = \frac{\alpha}{2}. \tag{4.2}$$

Řešení. Na Obrázku 4.1 je znázorněn graf hustoty $N(0,1)$ rozdělení a hodnoty x_α a $-x_\alpha$, které ohraničují vyznačené plochy, kde každá má obsah rovny $\frac{\alpha}{2}$.



Obrázek 4.1: Graf hustoty $N(0,1)$ rozdělení a hodnoty x_α a $-x_\alpha$, kde každá vymezuje plochu s obsahem $\frac{\alpha}{2}$.

Protože $\Pr[X > x_\alpha] = 1 - \Pr[X \leq x_\alpha]$ a to se rovná $1 - \Phi(x_\alpha)$, potom ze vztahu (4.2) dostáváme:

$$1 - \Phi(x_\alpha) = \frac{\alpha}{2} = 0,025.$$

Odtud dostaneme, že $\Phi(x_\alpha) = 1 - 0,025 = 0,975$. V statistické tabulce pro $N(0,1)$ rozdělení najdeme výslednou hodnotu $x_\alpha = 1,96$. ■

4.3 d-monomiální test

V této části se podíváme na d-monomiální test, v kterém testujeme, jestli ANF booleovské funkce má očekávaný počet monomů stupně $d \geq 1$. Budeme používat χ^2 test dobré shody a očekávaný počet monomů získáme z Věty 3.

V této podkapitole aplikujeme d-monomiální test na proudovou šifru. Nechť K je tajný klíč a IV inicializační vektor testované proudové šifry. Z IV si vybereme nějakých n složek $iv_1, \ldots, iv_n$ a budeme zkoumat booleovskou funkci $f(iv_1, \ldots, iv_n)$. Všechny ostatní bity inicializačního vektoru IV a všechny bity tajného klíče K jsou považovány za konstantní.

Pro všech 2^n možných vektorů $(iv_1, \ldots, iv_n)$ inicializujeme proudovou šifru pomocí K a IV a spočteme první bit keystreamu získaného ihned po inicializační fázi šifry. Získaný vektor délky 2^n si označme v. Potom pomocí Algoritmu 7 pro výpočet ANF spočteme z vektoru v algebraickou normální formu a výsledek uložíme zpět do vektoru v. Dále pro $d = 1, \ldots, n-1$ spočteme počet monomů stupně d algebraické normální formy, který si označíme m_d.

Podle Věty 3 má náhodná veličina popisující počet monomů stupně d ANF náhodné booleovské funkce normální rozdělení $N(\mu, \sigma^2)$ se střední hodnotou $\mu = \frac{1}{2}\binom{n}{d}$ a rozptylem $\sigma^2 = \frac{1}{4}\binom{n}{d}$. Dále aplikujeme χ^2 test dobré shody o $n-2$ stupních volnosti. Pro správné použití χ^2 testu dobré shody se doporučuje, aby platilo $n \geq 10$.

Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A následovně:

H_0 : naměřené hodnoty m_d se shodují s očekávanými hodnotami $\frac{1}{2}\binom{n}{d}$

H_A : naměřené hodnoty m_d se neshodují s očekávanými hodnotami $\frac{1}{2}\binom{n}{d}$.

Dále spočteme hodnotu testové statistiky:

$$X^2 = \sum_{d=1}^{n-1} \frac{\left(m_d - \frac{1}{2}\binom{n}{d}\right)^2}{\frac{1}{2}\binom{n}{d}}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy rozhodneme na základě kvantilu $\chi^2_{1-\alpha;\, n-2}$. Pracujeme na hladině významnosti $\alpha = 0,05$ a uvažujeme $n-2$ stupňů volnosti. Pokud $X^2 > \chi^2_{0,95;\, n-2}$, potom zamítneme nulovou hypotézu na hladině významnosti 0,05. Kvantil $\chi^2_{0,95;\, n-2}$ najdeme v příslušné tabulce kvantilů pro χ^2- rozdělení. Shrnutý d-monomiální test najdeme v Algoritmu 9.

Poznamenejme, že v článku [18] se hodnota testové statistiky počítá následovně:

$$X^2 = \sum_{d=0}^{n} \frac{\left(m_d - \frac{1}{2}\binom{n}{d}\right)^2}{\frac{1}{2}\binom{n}{d}}.$$

Avšak pro $d = 0$ nebo $d = n$ není splněna podmínka $\frac{1}{2}\binom{n}{d} \geq 5$, která se doporučuje pro správné použití χ^2 testu dobré shody.

Algoritmus 9 : d-monomiální test

for $iv = 0$ **to** $2^n - 1$ **do**
  inicializuj šifru s iv
  $v[iv]$ = první bit keystreamu
end for
spočítej ANF z vektoru v a výsledek ulož do v
for $i = 0$ **to** $2^n - 1$ **do**
  if $v[i] == 1$ **then**
    deg=stupeň monomu i
    distr[deg]=distr[deg]+1
end for
for $d = 1$ **to** $n - 1$ **do**
  $X^2 = X^2 + \frac{\left(\text{distr[d]} - \frac{1}{2}\binom{n}{d}\right)^2}{\frac{1}{2}\binom{n}{d}}$
end for
if $X^2 > \chi^2_{1-\alpha;\, n-2}$ **then**
  return šifra neprošla testem
else
  return šifra prošla testem

4.4 Monomiální distribuční test

Monomiální distribuční test je podobný d-monomiálnímu testu z předchozí podkapitoly s tím rozdílem, že místo jedné ANF uvažujeme P algebraických normálních forem a zkoumáme, v kolika z nich se každý monom vyskytuje.

Stejně jako v d-monomiálním testu získáme ANF na základě vybraných n složek $iv_1, \ldots, iv_n$ z inicializačního vektoru. Zbylých $P - 1$ ANF získáme analogicky, přičemž vždy vybereme jiných n složek z inicializačního vektoru, které dosud nebyly vybrány pro výpočet ANF.

Každou ANF si můžeme reprezentovat pomocí booleovského polynomu

$$a_0 \oplus a_1 x_1 \oplus \ldots \oplus a_n x_n \oplus a_{n+1} x_1 x_2 \oplus \ldots \oplus a_{2^n-1} x_1 x_2 \ldots x_n,$$

kde $a_i \in \mathbb{F}_2$. Nechť m_{a_i} značí, kolik z P testovaných booleovských polynomů má koeficient a_i nenulový. U náhodně zvoleného booleovského polynomu platí, že $\Pr[a_i = 1] = \frac{1}{2}$ pro $i = 0, \ldots, 2^n - 1$. Potom počet booleovských polynomů majících koeficient a_i rovny jedné popisuje náhodná veličina s binomickým rozdělením s parametry P a $p = \frac{1}{2}$, která má střední hodnotu rovnu $\frac{P}{2}$.

Analogicky jako v předchozím testu použijeme χ^2 test dobré shody o $2^n - 1$ stupních volnosti. Pro správné použití χ^2 testu dobré shody se doporučuje, aby platilo $\frac{P}{2} \geq 5$.

Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A následovně:

H_0 : naměřené hodnoty m_{a_i} se shodují s očekávanými hodnotami $\frac{P}{2}$

H_A : naměřené hodnoty m_{a_i} se neshodují s očekávanými hodnotami $\frac{P}{2}$.

Potom spočteme hodnotu testové statistiky:

$$X^2 = \sum_{i=0}^{2^n-1} \frac{\left(m_{a_i} - \frac{P}{2}\right)^2}{\frac{P}{2}}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy rozhodneme na základě kvantilu $\chi^2_{1-\alpha;\ 2^n-1}$. Pracujeme na hladině významnosti $\alpha = 0,05$, a uvažujeme $2^n - 1$ stupňů volnosti. Pokud $X^2 > \chi^2_{0,95;\ 2^n-1}$, potom zamítneme nulovou hypotézu na hladině významnosti 0,05. Kvantil $\chi^2_{0,95;\ 2^n-1}$ najdeme v příslušné tabulce kvantilů χ^2- rozdělení. Monomiální distribuční test je shrnut v Algoritmu 10.

Algoritmus 10 : Monomiální distribuční test

1: **for** $j = 1$ **to** P **do**
2: **for** $iv = 0$ **to** $2^n - 1$ **do**
3: inicializuj šifru s iv
4: $v[iv]$ = první bit keystreamu
5: **end for**
6: spočítej ANF z vektoru v a výsledek ulož do v
7: **for** $i = 0$ **to** $2^n - 1$ **do**
8: **if** $v[i] == 1$ **then**
9: $m_{a_i} = m_{a_i} + 1$
10: **end for**
11: **end for**
12: **for** $i = 0$ **to** $2^n - 1$ **do**
13: $X^2 = X^2 + \frac{\left(m_{a_i} - \frac{P}{2}\right)^2}{\frac{P}{2}}$
14: **end for**
15: **if** $X^2 > \chi^2_{1-\alpha;\ 2^n-1}$ **then**
 return šifra neprošla testem
16: **else**
 return šifra prošla testem

5. Strukturální analýza

V této kapitole se podíváme na čtyři strukurální testy z článku [19], které se používají k analýze synchronních proudových šifer. V prvním testu si vezmeme fixní inicializační vektor a budeme zkoumat korelaci mezi klíčem a odpovídajícím keystreamem. V druhém testu si zafixujeme klíč a budeme zkoumat korelaci mezi inicializačním vektorem a odpovídajícím keystreamem. V třetím testu uvažujeme různé inicializační vektory a podíváme se na korelaci mezi příslušnými keystreamy. V posledním testu této kapitoly budeme zkoumat difuzní vlastnosti každého bitu klíče a inicializačního vektoru.

Všechny čtyři testy jsou založené na χ^2-testu dobré shody, kde se používají kategorie, do kterých se přiřazují naměřené hodnoty. V popisech testů uvádíme konkrétně zvolené kategorie, které používáme v implementaci. Obecně můžeme definovat kategorie χ^2-testu dobré shody různými způsoby, postup testování se tím však nezmění.

V této kapitole budeme používat následující značení. Nechť S je proudová šifra, K je k-bitový klíč a IV je v-bitový inicializační vektor. Dále nechť z_i, kde $i = 1, 2, \ldots,$ značí keystream proudové šifry. Nakonec $S(K, IV, l)$ bude označovat prvních l bitů keystreamu vyprodukovaného šifrou S, která byla inicializována klíčem K a inicializačním vektorem IV.

5.1 Klíč/Keystream korelační test

V tomto testu zkoumáme pro fixní inicializační vektor korelaci mezi klíčem a odpovídajícím keystreamem. V článku [19, kap. 4.1] se uvádí, že pokud šifra „neprojde“ tímto testem, pak se doporučuje opravit část inicializační fáze šifry, v které se načítá klíč.

Na začátku testu se náhodně vygeneruje inicializační vektor IV a zafixuje se. Potom se m-krát náhodně vygeneruje k-bitový klíč. Pro každý klíč $K_i, i = 1, \ldots, m$ a fixní IV se vygeneruje keystream M_i délky k. Potom se každý klíč K_i po bitech naxoruje s příslušným keystreamem M_i a z výsledného binárního vektoru se spočte Hammingova váha w, jejíž hodnotu označíme $w_i = w\,(M_i \oplus K_i), i = 1, \ldots, m$. Poznamenejme, že pro bezpečnou šifru je rozdělení uvedených Hammingových vah w_i binomické s parametry $n = k$ a $p = \frac{1}{2}$ a platí

$$\Pr(\text{váha} = k') = \binom{n}{k'}(1/2)^n.$$

V testu uvažujeme délku klíče $k = 80$ a počet iterací $m = 2^{20}$. Dále uvažujme následujících pět kategorií navržených v [19, kap. 4.1]:

$$\text{(i) } 0 - 35,\ \text{(ii) } 36 - 38,\ \text{(iii) } 39 - 41,\ \text{(iv) } 42 - 44,\ \text{(v) } 45 - 80.$$

Každou váhu $w_i, i = 1, \ldots, m$, umístíme do příslušné kategorie. Pro každou kategorii spočteme počet jejích prvků, čímž dostaneme naměřené hodnoty, které pak v testu porovnáme s očekávanými hodnotami. Nechť X_1 označuje počet vah w_i, které patří do kategorie (i). Dále nechť X_2 označuje počet vah w_i, které patří do kategorie (ii), atd. až nechť X_5 označuje počet vah w_i, které patří do kategorie (v).

Nejprve si pro jednotlivé kategorie spočteme, s jakou pravděpodobností patří Hammingova váha w_i, u které předpokládáme, že má binomické rozdělení s parametry $n = 80$ a $p = \frac{1}{2}$, do dané kategorie.

$$\begin{aligned}
P_1 &= \Pr(0 \leq \text{váha} \leq 35) = \sum_{i=0}^{35} \frac{\binom{80}{i}}{2^{80}} \\
P_2 &= \Pr(36 \leq \text{váha} \leq 38) = \sum_{i=36}^{38} \frac{\binom{80}{i}}{2^{80}} \\
P_3 &= \Pr(39 \leq \text{váha} \leq 41) = \sum_{i=39}^{41} \frac{\binom{80}{i}}{2^{80}} \\
P_4 &= \Pr(42 \leq \text{váha} \leq 44) = \sum_{i=42}^{44} \frac{\binom{80}{i}}{2^{80}} \\
P_5 &= \Pr(45 \leq \text{váha} \leq 80) = \sum_{i=45}^{80} \frac{\binom{80}{i}}{2^{80}}.
\end{aligned}$$

Po zaokrouhlení dostaneme následující hodnoty pravděpodobností:

$$P_1 = 0,157, \quad P_2 = 0,212, \quad P_3 = 0,262, \quad P_4 = 0,212 \quad \text{a} \quad P_5 = 0,157.$$

Dále spočteme očekávané počty $E_i = m \cdot P_i$ prvků v jednotlivých kategoriích pro $i = 1, \ldots, 5$. Pro $m = 2^{20}$ dostaneme následující hodnoty:

$$E_1 = 164787, \quad E_2 = 221904, \quad E_3 = 275194, \quad E_4 = 221904 \quad \text{a} \quad E_5 = 164787,$$

které jsou nezávislé na volbě konkrétní šifry.

Předpokládejme, že máme naměřené hodnoty $X_1, \ldots, X_5$ pro konkrétní šifru. Potom aplikujeme χ^2 test dobré shody. Definujme si nulovou hypotézu H_0 a alternativní hypotézu H_A :

H_0: naměřené hodnoty X_i se shodují s očekávanými hodnotami E_i

H_A: naměřené hodnoty X_i se neshodují s očekávanými hodnotami E_i.

Dále spočteme hodnotu testové statistiky:

$$X^2 = \sum_{i=1}^{5} \frac{(X_i - E_i)^2}{E_i}.$$

O zamítnutí nebo nezamítnutí nulové hypotézy H_0 můžeme rozhodnout na základě p-hodnoty nebo kvantilu $\chi^2_{1-\alpha;\, f}$.

Pracujeme na hladině významnosti $\alpha = 0,05$ a uvažujeme $f = 4$ stupně volnosti. Kvantil $\chi^2_{1-\alpha;\, f}$ je pak rovný $\chi^2_{0,95;\, 4} = 9,488$. Pokud $X^2 > \chi^2_{0,95;\, 4}$, potom zamítneme nulovou hypotézu H_0 na hladině významnosti 0,05.

Jiný způsob, podle kterého rozhodujeme o zamítnutí nebo nezamítnutí nulové hypotézy, je založen na p-hodnotě. Na základě hodnoty X^2 spočteme p-hodnotu podle následujícího vzorce[1]:

$$\text{p-hodnota} = 1 - F(X^2),$$

[1]odůvodnění vzorce: rozdílné hodnoty X_i a E_i odporují hypotéze H_0 a způsobují vysokou hodnotu X^2. Čím je hodnota X^2 vyšší, tím více odporuje hypotéze H_0. Proto se p-hodnota spočítá jako $\Pr(T \geq X^2)$, kde T je náhodná veličina s χ^2-rozdělením o f stupních volnosti.

kde F je distribuční funkce χ^2-rozdělení o f stupních volnosti. Potom na základě spočtené p-hodnoty rozhodneme o zamítnutí nebo nezamítnutí nulové hypotézy podle tabulky (1.2) z první kapitoly této práce. Klíč/Keystream korelační test je shrnut v Algoritmu 11.

Algoritmus 11 : Klíč/Keystream test

VSTUP: počet iterací m
VÝSTUP: p-hodnota

1: náhodně vygeneruj a zafixuj inicializační vektor IV
2: **for** $i = 1$ **to** m **do**
3: náhodně vygeneruj klíč K délky k bitů
4: vygeneruj prvních k bitů keystreamu $M = S(K, IV, k)$
5: spočítej váhu $w_i = w(M \oplus K)$
6: **end for**
7: kategorizuj váhy w_i
8: aplikuj χ^2 test dobré shody:
spočítej: očekávané hodnoty $E_i = m.P_i$ pro $i = 1, \ldots, 5$
naměřené hodnoty $X_i, i = 1, \ldots, 5$
$X^2 = \sum_{i=1}^{5} \frac{(X_i - E_i)^2}{E_i}$
p-hodnotu
9: **return** p-hodnota

5.2 Inicializační vektor/Keystream korelační test

V tomto testu postupujeme analogickým způsobem jako v předchozím Klíč/Keystream korelačním testu, ale s tím rozdílem, že teď zafixujeme klíč K a zkoumáme korelaci mezi inicializačním vektorem IV a prvními v bity keystreamu. V článku [19, kap. 4.2] se uvádí, že pokud šifra „neprojde“ tímto testem, pak se doporučuje opravit část inicializační fáze šifry, v které se načítá IV. Na začátku testu se náhodně vygeneruje klíč K a zafixuje se. Potom se m-krát náhodně vygeneruje v-bitový inicializační vektor IV. Pro každý inicializační vektor $IV_i, i = 1, \ldots, m$ a fixní klíč K se vygeneruje keystream M_i délky v. Potom se každý IV_i po bitech naxoruje s příslušným keystreamem M_i a z výsledného binárního vektoru se spočte Hammingova váha $w_i = w(M_i \oplus IV_i), i = 1, \ldots, m$. Pro bezpečnou šifru je rozdělení uvedených Hammingových vah w_i binomické s parametry $n = v$ a $p = \frac{1}{2}$.

V testu uvažujeme délku inicializačního vektoru $v = 64$ a počet iterací $m = 2^{20}$. Dále uvažujme následujících pět kategorií navržených v [19, kap. 4.2]:

$$\text{(i) } 0 - 28, \text{ (ii) } 29 - 30, \text{ (iii) } 31 - 33, \text{ (iv) } 34 - 35, \text{ (v) } 36 - 64.$$

Každou váhu $w_i, i = 1, \ldots, m$, umístíme do příslušné kategorie. Pro každou kategorii spočteme počet její prvků, čímž dostaneme naměřené hodnoty X_i, které pak v testu porovnáme s očekávanými hodnotami E_i. Nejprve si pro jednotlivé kategorie spočteme, s jakou pravděpodobností patří Hammingova váha w_i, u které předpokládáme binomické rozdělení s parametry $n = 64$ a $p = \frac{1}{2}$, do dané kategorie.

$$
\begin{aligned}
P_1 &= \Pr(0 \leq \text{váha} \leq 28) = \sum_{i=0}^{28} \frac{\binom{64}{i}}{2^{64}} \\
P_2 &= \Pr(29 \leq \text{váha} \leq 30) = \sum_{i=29}^{30} \frac{\binom{64}{i}}{2^{64}} \\
P_3 &= \Pr(31 \leq \text{váha} \leq 33) = \sum_{i=31}^{33} \frac{\binom{64}{i}}{2^{64}} \\
P_4 &= \Pr(34 \leq \text{váha} \leq 35) = \sum_{i=34}^{35} \frac{\binom{64}{i}}{2^{64}} \\
P_5 &= \Pr(36 \leq \text{váha} \leq 64) = \sum_{i=36}^{64} \frac{\binom{64}{i}}{2^{64}}.
\end{aligned}
$$

Po zaokrouhlení dostaneme:

$$P_1 = 0,191, \quad P_2 = 0,163, \quad P_3 = 0,292, \quad P_4 = 0,163 \quad \text{a} \quad P_5 = 0,191.$$

Očekávané počty $E_i = m \cdot P_i$ prvků v jednotlivých kategoriích jsou pro $m = 2^{20}$ následující:

$$E_1 = 200278, \quad E_2 = 170918, \quad E_3 = 306184, \quad E_4 = 170918 \quad \text{a} \quad E_5 = 200278.$$

Předpokládejme, že máme naměřené hodnoty $X_1, \ldots, X_5$ pro konkrétní šifru. Stejným způsobem jako v předchozím Klíč/Keystream korelačním testu aplikujeme χ^2 test dobré shody. Protože se jedná o identický postup testování, nebudeme ho znovu uvádět.

Inicializační vektor/Keystream korelační test je shrnut v Algoritmu 12.

Algoritmus 12 : IV/Keystream test

VSTUP: počet iterací m
VÝSTUP: p-hodnota

1: náhodně vygeneruj a zafixuj klíč K
2: **for** $i = 1$ **to** m **do**
3: náhodně vygeneruj inicializační vektor IV délky v bitů
4: vygeneruj prvních v bitů keystreamu $M = S(K, IV, v)$
5: spočítej váhu $w_i = w(M \oplus IV)$
6: **end for**
7: kategorizuj váhy w_i
8: aplikuj χ^2 test dobré shody
9: **return** p-hodnota

5.3 Korelační test pro rámec

V tomto testu zkoumáme korelaci mezi rámci[2], které byly vygenerovány pomocí podobných inicializačních vektorů. V článku [19, kap. 4.3] se uvádí, že pokud šifra „neprojde“ tímto testem, pak se doporučuje opravit část inicializační fáze šifry, v které se načítá IV.

Na začátku testu si náhodně vygenerujeme klíč K a inicializační vektor IV. Potom m-krát na základě K a IV vyprodukujeme pomocí testované šifry keystream délky l, přičemž v každé iteraci po vygenerování keystreamu inkrementujeme IV. Jednotlivých m keystreamů si uložíme po řádcích do matice o rozměrech $m \times l$. Dále pro každý sloupec matice spočteme jeho Hammingovu váhu $w_i, i = 1, \ldots, l$. Pro bezpečnou šifru je rozdělení uvedených Hammingových vah w_i binomické s parametry $n = m$ a $p = \frac{1}{2}$.

V testu uvažujeme počet iterací $m = 2^{20}$ a délku keystreamu $l = 192$. Podobně jako v předchozích dvou testech i v tomto testu uvažujeme pět kategorií pro váhy w_i, které si označíme následujícím způsobem:

(i) $0 - x_1$, (ii) $(x_1 + 1) - x_2$, (iii) $(x_2 + 1) - x_3$, (iv) $(x_3 + 1) - x_4$, (v) $(x_4 + 1) - m$.

Protože pro tento test nebyly v článku [19] specifikované jednotlivé kategorie(ani jejich počet), navrhneme vlastní kategorie, které uvedeme v příkladě na konci této podkapitoly.

Každou váhu $w_i, i = 1, \ldots, l$, umístíme do příslušné kategorie. Pro každou kategorii spočteme počet její prvků, čímž dostaneme naměřené hodnoty X_i, které pak v testu porovnáme s očekávanými hodnotami E_i. Potom stejným způsobem jako u předchozích dvou testů této kapitoly aplikujeme χ^2 test dobré shody.

Korelační test pro rámec je shrnut v Algoritmu 13.

Algoritmus 13 : Korelační test pro rámec

VSTUP: počet iterací m, délka keystreamu l
VÝSTUP: p-hodnota

1: náhodně vygeneruj K a IV
2: **for** $i = 1$ **to** m **do**
3: vygeneruj prvních l bitů keystreamu $M_i = (M_{i1}, \ldots, M_{il}) = S(K, IV, l)$
4: inkrementuj IV
5: **end for**
6: **for** $j = 1$ **to** l **do**
7: $w_j = \sum_{i=1}^{m} M_{ij}$
8: **end for**
9: kategorizuj váhy w_i
10: aplikuj χ^2 test dobré shody
11: **return** p-hodnota

[2]rámec(frame) označuje posloupnost bitů keystreamu pevné délky, která byla vygenerována s použitím jednoho inicializačního vektoru. Pro vygenerování následujícího rámce se používá jiný inicializační vektor, který obvykle vznikne jako inkrementace inicializačního vektoru z předchozího rámce

Příklad. V tomto příkladě si spočteme hodnoty $x_1, \ldots, x_5$, pomocí kterých jsou definovány kategorie (i) až (v) pro Korelační test pro rámec. Při výpočtu hodnot $x_1, \ldots, x_5$ budeme předpokládat, že váha w_i binárního vektoru délky n má binomické rozdělení s parametry $n = 2^{20}$ a $p = \frac{1}{2}$. Hodnoty $x_1, \ldots, x_5$ si zvolíme tak, aby všechny kategorie měly přibližně stejnou pravděpodobnost, že do nich padne váha w_i, kde $i \in \{1, \ldots, l\}$.

Řešení. Pro určení hodnot $x_1, \ldots, x_5$ využijeme Centrální limitní větu pro náhodnou veličinu s binomickým rozdělením. Označme si náhodnou veličinu Y s binomickým rozdělením s parametry $n = 2^{20}$ a $p = \frac{1}{2}$, která popisuje váhy w_i. Potom náhodná veličina Y má střední hodnotu $np = n/2$ a rozptyl $np(1-p) = n/4$.

Nejprve spočteme hraniční hodnotu x_1. Hledáme x_1 takové, že

$$\Pr(\text{váha} \leq x_1) = \Pr(Y \leq x_1) = \frac{1}{5}.$$

Použijeme Centrální limitní větu pro náhodnou veličinu s binomickým rozdělením a získáme

$$\Pr(Y \leq x_1) = \Pr\left(\frac{Y - \frac{n}{2}}{\sqrt{\frac{n}{4}}} \leq \frac{x_1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right) \underset{n\to\infty}{\longrightarrow} \Phi\left(\frac{x_1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right).$$

Potom dostaneme, že $\Phi\left(\frac{x_1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right)$ je přibližně rovna $\frac{1}{5}$ a po úpravě získáme vztah $\Phi\left(\frac{2x_1 - n}{\sqrt{n}}\right) \approx \frac{1}{5}$. Z vlastností distribuční funkce normovaného normálního rozdělení víme, že

$$\Phi(-x) = 1 - \Phi(x).$$

Označíme si $x = \frac{2x_1 - n}{\sqrt{n}}$, potom platí $\Phi(-x) = \Phi\left(\frac{n - 2x_1}{\sqrt{n}}\right) \approx \frac{4}{5}$. V příslušné statistické tabulce pro $N(0,1)$ najdeme, že $\Phi^{-1}\left(\frac{4}{5}\right) = 0,841$ a dostáváme vztah $\frac{n - 2x_1}{\sqrt{n}} \approx 0,841$. Elementárními úpravami získáme $x_1 \approx \frac{n - 0,841.\sqrt{n}}{2}$ a pro $n = 2^{20}$ dostaneme $x_1 \approx 523857$.

Podobným způsobem spočteme hraniční hodnotu x_2, pro kterou platí:

$$\Pr(x_1 + 1 \leq \text{váha} \leq x_2) = \frac{1}{5}$$

Podle Centrální limitní věty pro náhodnou veličinu s binomickým rozdělením získáme:

$$\frac{1}{5} = \Pr\left(\frac{x_1 + 1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}} \leq \frac{Y - \frac{n}{2}}{\sqrt{\frac{n}{4}}} \leq \frac{x_2 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right) \approx \Phi\left(\frac{x_2 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right) - \Phi\left(\frac{x_1 + 1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right).$$

Pro dané x_1 a n se $\Phi\left(\frac{x_1 + 1 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right)$ přibližně rovná 0,201. Potom dostáváme vztah $\Phi\left(\frac{x_2 - \frac{n}{2}}{\sqrt{\frac{n}{4}}}\right) \approx 0,401$, z kterého analogickými úvahami jako u výpočtu x_1 získáme hodnotu x_2.

Stejným způsobem najdeme hraniční hodnoty x_3 a x_4. Potom dostáváme následující kategorie:

(i) $0 - 523857$, (ii) $523858 - 524159$, (iii) $524160 - 524418$,

(iv) $524419 - 524721$, (v) $524722 - 2^{20}$.

Očekávané počty $E_i = l \cdot \frac{1}{5}$ prvků v jednotlivých kategoriích jsou pro $l = 192$ rovny $E_i = 38,4$, kde $i = 1, \ldots, 5$. ■

5.4 Difuzní test

Tento test zkoumá difuzi každého bitu klíče K a každého bitu inicializačního vektoru IV na keystreamu. Na začátku testu si náhodně vygenerujeme klíč K délky k a inicializační vektor IV délky v a na základě nich testovaná šifra S vyprodukuje l bitů keystreamu $S(K, IV, l)$. Potom postupně pro všechny bity klíče a inicializačního vektoru v každé iteraci změníme jeden bit a vygenerujeme nový keystream délky l, který naxorujeme s původním keystreamem $S(K, IV, l)$. Výsledné posloupnosti bitů budeme po řádcích ukládat do matice, která bude typu $(k + v) \times l$. Celou uvedenou procedúru vykonáme m-krát a získaných m matic navzájem sčítáme do jedné výsledné matice M. V článku [19, kap. 4.4] se uvádí, že pokud šifra „neprojde“ tímto testem, pak se doporučuje opravit inicializační část testované šifry.

Pro bezpečnou šifru očekáváme, že každý prvek matice M má binomické rozdělení s parametry $n = m$ a $p = \frac{1}{2}$. Prvky matice M přiřadíme do pevně zvolených kategorií. V článku [19, kap. 4.4] pro $m = 1024$ bylo navržených následujících pět kategorií:

(i) $0 - 498$, (ii) $499 - 507$, (iii) $508 - 516$, (iv) $517 - 525$ a (v) $526 - 1024$.

Pro jednotlivé kategorie si spočteme s jakou pravděpodobností patří prvek matice M, u kterého předpokládáme binomické rozdělení s parametry $n = 1024$ a $p = \frac{1}{2}$, do dané kategorie:

$$\begin{aligned}
P_1 &= \Pr(0 \leq \text{váha} \leq 498) = \sum_{i=0}^{498} \frac{\binom{1024}{i}}{2^{1024}} \\
P_2 &= \Pr(499 \leq \text{váha} \leq 507) = \sum_{i=499}^{507} \frac{\binom{1024}{i}}{2^{1024}} \\
P_3 &= \Pr(508 \leq \text{váha} \leq 516) = \sum_{i=508}^{516} \frac{\binom{1024}{i}}{2^{1024}} \\
P_4 &= \Pr(517 \leq \text{váha} \leq 525) = \sum_{i=517}^{525} \frac{\binom{1024}{i}}{2^{1024}} \\
P_5 &= \Pr(526 \leq \text{váha} \leq 1024) = \sum_{i=526}^{1024} \frac{\binom{1024}{i}}{2^{1024}}.
\end{aligned}$$

Po zaokrouhlení dostaneme:

$$P_1 = 0,199; \quad P_2 = 0,190; \quad P_3 = 0,222; \quad P_4 = 0,190 \quad \text{a} \quad P_5 = 0,199.$$

Očekávané počty $E_i = (k+v) \cdot l \cdot P_i$ prvků v jednotlivých kategoriích jsou pro $k = 80, v = 64$ a $l = 192$ následující:

$$E_1 = 195625, \quad E_2 = 186778, \quad E_3 = 218234, \quad E_4 = 186778 \quad \text{a} \quad E_5 = 195625.$$

Předpokládejme, že máme naměřené hodnoty $X_1, \ldots, X_5$ pro konkrétní šifru. Potom aplikujeme χ^2 test dobré shody stejným způsobem jako u předchozích testů této kapitoly. Pseudokód[3] Difuzního testu je obsažen v Algoritmu 14.

Algoritmus 14 : Algoritmus Difuzní test

VSTUP: počet iterací m
VÝSTUP: p-hodnota

1: vygeneruj nulovou matici M o rozměrech $(k+v) \times l$

$$M = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{k+v} \end{bmatrix}$$

2: **for** $i = 1$ **to** m **do**
3: zvol náhodný K a IV
4: $C = S(K, IV, l)$
5: **for** $j = 1$ **to** k **do**
6: $K' = K \oplus e_j$ {e_j je vektor samých nul a jedné jedničky na pozici j}
7: $D = C \oplus S(K', IV, l)$
8: $M_j = M_j + D$
9: **end for**
10: **for** $j = 1$ **to** v **do**
11: $IV' = IV \oplus e_j$
12: $D = C \oplus S(K, IV', l)$
13: $M_{k+j} = M_{k+j} + D$
14: **end for**
15: **end for**
16: kategorizuj váhy w_i
17: aplikuj χ^2 test dobré shody
18: **return** p-hodnota

[3] e_j označuje v pseudokódu binární vektor příslušné délky, který na j-té pozici má jedničku a všude jinde má nuly

6. Využití PRBG v Pollardově ρ metodě

V této kapitole si uvedeme Pollardovu ρ metodu, která slouží pro faktorizaci složených čísel. Dále si představíme způsob využití generátorů pseudonáhodných bitů pro faktorizaci čísel pomocí Pollardovy ρ metody.

6.1 Pollardova ρ metoda

Pollardova ρ metoda je pravděpodobnostní metoda pro faktorizaci složeného čísla, která byla navržená J.M.Pollardem v roce 1975. V algoritmu sestrojíme posloupnost čísel, která se od určitého indexu zacyklí. Svým provedením algoritmus připomíná řecké písmeno ρ, podle kterého bývá tato metoda nazývaná. Více podrobností o Pollardově ρ metodě najdeme v [20, kap. 8.5] a [21].

Nechť máme složené číslo $n \in \mathbb{N}$, které chceme faktorizovat. Zvolíme si náhodný polynom $f \in \mathbb{Z}_n[x]$ a náhodně si zvolíme inicializační prvek $x_0 \in \mathbb{Z}_n$. Základním prvkem Pollardovy ρ metody je posloupnost pseudonáhodných čísel, kterou můžeme zkonstruovat následujícím způsobem:

$$x_{i+1} = f(x_i) \bmod n \text{ pro } i \geq 0.$$

Množina $\mathbb{Z}_n$ je konečná, a proto musí v posloupnosti vzniknout cyklus. Pro praktické využití algoritmu se doporučuje [3, kap. 3.2.2] volit $x_0 = 2$ a $f(x) = x^2 + c$, kde $c \neq 0, -2$.

Nechť n je složené číslo a předpokládejme, že q a t, přičemž $0 < q \leq t$, jsou netriviální dělitelé n, pro které platí $n = qt$. Dále předpokládejme, že jsme našli taková nezáporná celá čísla i a $j, i < j$, pro které platí, že $x_i \equiv x_j \bmod q$ a $x_i \not\equiv x_j \bmod n$. Potom platí, že $q|(x_i - x_j)$. Protože $q|n$ a $q|(x_i - x_j)$, potom $q|\text{NSD}(n, x_i - x_j)$. Předpokládali jsme, že q je netriviální dělitel čísla n, tj. $q \geq 2$, potom $\text{NSD}(n, x_i - x_j) \geq 2$. Víme, že $\text{NSD}(n, x_i - x_j) > 1$ a $\text{NSD}(n, x_i - x_j)|n$. Podle předpokladu $n \nmid (x_i - x_j)$. Odsud dostáváme, že $n \nmid \text{NSD}(n, x_i - x_j)$ a $\text{NSD}(n, x_i - x_j)$ je netriviální dělitel čísla n.

Netriviální dělitelé q a t zatím neznáme. Posloupnost x_i využijeme k nalezení netriviálního dělitele čísla n. V každém kroku algoritmu spočteme $\text{NSD}(n, x_i - x_j)$. Připomeňme, že potřebujeme najít taková i a j, že platí $x_i \equiv x_j \bmod q$ a $x_i \not\equiv x_j \bmod n$. Poznamenejme, že posloupnost $x_i \bmod q$ je periodická.

Pollard navrhl používat Floydův algoritmus na hledání cyklu, díky kterému dosáhneme snížení počtu kroků potřebných pro nalezení takových celých čísel, že platí $x_i \equiv x_j \bmod q$. Pokud platí, že $x_i \equiv x_j \bmod q$, potom platí $x_{i+1} \equiv x_{j+1} \bmod q$. Jestliže budeme pokračovat v sestrojení posloupnosti $x_{i+k} \equiv x_{j+k} \bmod q$, kde $k = 1, \ldots$, tak časem najdeme i takové, že $x_{2i} \equiv x_i \bmod q$. Použijeme pomocnou proměnnou y_0 a do ní dosadíme náhodně zvolený inicializační prvek x_0. Dále rekurzivně definujeme $y_{i+1} = f(f(y_i))$ pro $i \geq 0$. Získáme následující rovnosti:

$$
\begin{aligned}
y_0 &= x_0 \\
y_1 &= x_2 = f(x_1) = f(f(x_0)) = f(f(y_0)) \\
y_2 &= x_4 = f(x_3) = f(f(x_2)) = f(f(y_1)) \\
&\vdots \\
y_i &= x_{2i} = f(x_{2i-1}) = f(f(x_{2i-2})) = f(f(y_{i-1})).
\end{aligned}
$$

Pro každé i spočteme $\mathrm{NSD}(n, |x_i - x_{2i}|)$ pouze jednou.

Dále uvedeme pseudokód Pollardovy ρ metody. Na vstupu očekáváme složené číslo $n \in \mathbb{N}$ a výstupem z algoritmu je vlastní dělitel d čísla n nebo neúspěch.

Algoritmus 15 : Pollardova ρ metoda

VSTUP: složené číslo $n \in \mathbb{N}$
VÝSTUP: netriviální dělitel d nebo neúspěch

1: náhodně zvol $f \in \mathbb{Z}_n[x]$
2: náhodně zvol inicializační prvek $x_0 \in \mathbb{Z}_n$
3: $x = x_0$, $y = x_0$, $d = 1$
4: **while** $d == 1$ **do**
5: $\quad x = f(x) \mod n$
6: $\quad y = f(f(y)) \mod n$
7: $\quad$ spočítej $d = \mathrm{NSD}(n, |x - y|)$
8: **end while**
9: **if** $d < n$ **then**
$\quad$ **return** d
10: **else**
$\quad$ **return** neúspěch

Pokud Pollardova ρ metoda skončí neúspěchem, tak se doporučuje zvolit jiný polynom $f \in \mathbb{Z}_n[x]$.

Příklad. Pomocí Pollardovy ρ metody faktorizujte složené číslo $n = 1357$ s využitím polynomu $f(x) = x^2 + 1 \in \mathbb{Z}_n[x]$ a inicializačního prvku $x_0 = 1$.

Řešení. Hledáme index i takový, že $\mathrm{NSD}(n, |x_i - y_i|) = d$, kde $y_i = x_{2i}$ a $1 < d < n$. Generujeme posloupnost $x_1, x_2, \ldots$ pomocí následujícího vztahu:

$$x_{i+1} = f(x_i) \bmod n \text{ pro } i \geq 0$$

a spočteme NSD pro všechny iterace.

$$
\begin{aligned}
i = 0: \quad & x_0 = 1 \\
& y_0 = 1 \\
& d = 1
\end{aligned}
$$

$$
\begin{aligned}
i = 1: \quad & x_1 = f(x_0) = f(1) = 2 \\
& y_1 = f(f(y_0)) = f(f(1)) = 5 \\
& d = \mathrm{NSD}(1357, |2 - 5|) = 1
\end{aligned}
$$

$$
\begin{aligned}
i = 2: \quad & x_2 = f(x_1) = f(2) = 5 \\
& y_2 = f(f(y_1)) = f(f(5)) = 677 \\
& d = \mathrm{NSD}(1357, |5 - 677|) = 1
\end{aligned}
$$

$$
\begin{aligned}
i = 3: \quad & x_3 = f(x_2) = f(5) = 26 \\
& y_3 = f(f(y_2)) = f(f(677)) = 266 \\
& d = \mathrm{NSD}(1357, |26 - 266|) = 1 \\
\\
i = 4: \quad & x_4 = f(x_3) = f(26) = 677 \\
& y_4 = f(f(y_3)) = f(f(26)) = 611 \\
& d = \mathrm{NSD}(1357, |677 - 611|) = 1 \\
\\
i = 5: \quad & x_5 = f(x_4) = f(677) = 1021 \\
& y_5 = f(f(y_4)) = f(f(611)) = 1255 \\
& d = \mathrm{NSD}(1357, |1021 - 1255|) = 1 \\
\\
i = 6: \quad & x_6 = f(x_5) = f(1021) = 266 \\
& y_6 = f(f(y_5)) = f(f(1255)) = 1209 \\
& d = \mathrm{NSD}(1357, |266 - 1209|) = 23
\end{aligned}
$$

Výstupem z algoritmu je číslo 23. Potom dostáváme, že původní číslo 1357 se rovná součinu čísel 23 a 59. ■

6.2 Využití PRBG v Pollardově ρ metodě

V této části si uvedeme postup využití generátorů pseudonáhodních bitů pro faktorizaci čísel pomocí Pollardovy ρ metody. Mezi základní části Pollardovy ρ metody patří pseudonáhodná posloupnost čísel konstruována pomocí vztahu $x_{i+1} = f(x_i) \mod n$, kde f je polynom s celočíselnými koeficienty. Hlavní idea spočívá v tom, že v Pollardově ρ metodě místo polynomu f použijeme generátor pseudonáhodných bitů. Pseudokód Pollardovy ρ metody s využitím PRBG bude podobný jako v Algoritmu 15 s tím rozdílem, že místo výrazu "$f(x) \mod n$" budeme využívat výraz "$\mathrm{PRBG}(x) \mod n$", který má následující význam. Hodnotu $x \in \mathbb{Z}_n$ převedeme do binárního tvaru a chápeme ji jako semínko PRBG. PRBG potom vygeneruje posloupnost bitů dostatečné délky(např. $\lfloor \log_2 n \rfloor + 1$), která je pak převedena do dekadického tvaru a spočtena modulo n.

Poznamenejme však, že čísla vygenerováná výrazem "$\mathrm{PRBG}(x) \mod n$" nebudou mít diskrétní rovnoměrné rozdělení na množině $\{0, \ldots, n-1\}$. Čísla s nižšími hodnotami budou pravděpodobnejší, než čísla z vyššími hodnotami, protože když se vygenerovaná posloupnost pseudonáhodných bitů převede do dekadického tvaru, může nabývat vyšší hodnoty než dané n a tehdy se použije operace modulo n. V implementační části otestujeme, jaký vliv bude mít toto pozorování na úspešnost Pollardovy ρ metody.

V následující kapitole otestujeme, jestli uvedené generátory pseudonáhodných bitů Blum Blum Shub generátor, Geffe generátor a Decim jsou vhodné pro účely faktorizace pomocí Pollardovy ρ metody.

7. Výsledky testování

V práci jsme naimplementovali následující testy:

Frekvenční monobit test	Alg. 3	Monomiální distribuční test	Alg. 10
Frekvenční monobit test (mod.)	Alg. 4	Klíč/Keystream test	Alg. 11
Modifikovaný Poker test	Alg. 5	IV/Keystream test	Alg. 12
Runs test	Alg. 6	Korelační test pro rámec	Alg. 13
Afinní konstantní test	Alg. 8	Difuzní test	Alg. 14
d-monomiální test	Alg. 9		

Uvedené testy jsme aplikovali na generátory Blum Blum Shub, Geffe a Decim. Před samotným představením výsledků testování si nejdřív uvedeme poznámky k implementaci.

7.1 Poznámky k implementaci

Testy i generátory jsme naprogramovali v jazyce Java. Pro spuštění jednotlivých programů je nutné mít nainstalovanou Javu JRE 8. Implementaci jsme rozdělili do následujících balíků:

- *prbg* - obsahuje implementaci generátorů BBS, Geffe a Decimu
- *helpFiles* - pomocný balík pro práci se soubory
- *basicTest* - obsahuje implementaci základních statistických testů: Frekvenční monobit test, modifikace Frekvenčního monobit testu, Modifikovaný Poker test a Runs test
- *monomialTests* - obsahuje implementaci Monomiálních testů: Afinní konstantní test, d-monomiální test a Monomiální distribuční test
- *StructuralTests* - obsahuje implementaci Strukturální analýzy: Klíč/Keystream test, IV/Keystream test, Korelační test pro rámec a Difuzní test
- *pollardRho* - obsahuje implementaci Pollardovy ρ metody a implementaci tříd pro využití PRBG v Pollardově ρ metodě

Naše implementace používá externí knihovnu *org.apache.commons.math3* [22], z které používáme implementaci(třída *distribution.NormalDistribution*) distribuční funkce Φ normálního rozdělení.

Parametry počítače, který jsme použili k testování, jsou následující:

- Procesor: Intel Xeon CPU E1245 @ 3,30GHz, x64
- RAM: 12,0GB
- OS: Windows 7, 64 bit.

Dále si specifikujeme parametry pro Geffe generátor a pro Pollardovu ρ metodu, které jsme použili v implementaci.

Geffe generátor - specifikace parametrů

Nechť L_1, L_2 a L_3 označují tři LFSR Geffe generátoru. Pro naši implementaci Geffe generátoru jsme si zvolili následující charakteristické polynomy ze $\mathbb{Z}_2[x]$, které jsme převzali z článku [8]:

$L_1 : 1 + x^{12} + x^{17}$, kde délka L_1 je 17
$L_2 : 1 + x^{18} + x^{25}$, kde délka L_1 je 25
$L_3 : 1 + x^6 + x^7$, kde délka L_1 je 7

Každý z uvedených charakteristických polynomů je primitivní.

Způsob, jakým se inicializují LFSR pomocí klíče K a inicializačního vektoru IV pro Geffe generátor, jsme nenalezli v žádné literatuře. Inspirovali jsme se inicializačním algoritmem pro proudovou šifru A5/1 [23] a pro Geffe generátor jsme použili následující inicializační algoritmus.

Algoritmus 16 : Inicializační algoritmus

VSTUP: K-klíč, IV-inicializační vektor

1: nastav všechny políčka tří registrů na nulu
2: **for** $i = 1$ **to** délka klíče K **do**
3: $L_1[0] = L_1[0] \oplus K[i]$
4: $L_2[0] = L_2[0] \oplus K[i]$
5: $L_3[0] = L_3[0] \oplus K[i]$
6: všechny tři LFSR se posunou
7: **end for**
8: **for** $i = 1$ **to** délka rámce M **do**
9: $L_1[0] = L_1[0] \oplus IV[i]$
10: $L_2[0] = L_2[0] \oplus IV[i]$
11: $L_3[0] = L_3[0] \oplus IV[i]$
12: všechny tři LFSR se posunou
13: **end for**

Délku klíče K jsme zvolili 80 bitů a délku inicializačního vektoru IV jsme zvolili 64 bitů.

Pollardova ρ metoda - specifikace parametrů

V implementaci Pollardovy ρ metody jsme pro dané n používali polynom $f(x) = x^2 + 1 \in \mathbb{Z}_n[x]$ a inicializační prvek $x_0 = 1$.

Implementace PRBG v Pollardově ρ metodě

V Pollardově ρ metodě jsme místo polynomu $f \in \mathbb{Z}_n[x]$ pro výpočet $f(x)$ mod n použili Geffe generátor a Decim následujícím způsobem. Nejdřív si $x \in \mathbb{Z}_n$ převedeme do binárního tvaru a v případě potřeby doplníme nulami, aby jsme dostali binární vektor délky 80 bitů, který budeme používat jako klíč K. Potom si pro klíč K a nulový inicializační vektor spočteme 64 bitů keystreamu, které pak zpět převedeme do dekadického tvaru a spočteme modulo n.

Blum Blum Shub generátor jsme v Pollardově ρ metodě nepoužili, protože není jasné, jakým způsobem by se mohl v této metodě implementovat.

7.2 Naměřené výsledky

V této podkapitole si uvedeme naměřené výsledky, které jsme získali aplikováním uvedených testů na generátory BBS, Geffe a Decim.

Základní statistické testy

Generátory BBS, Geffe a Decim jsme otestovali pomocí základních statistických testů: Frekvenční monobit test, modifikace Frekvenčního monobit testu, Modifikovaný Poker test a Runs test. Každým generátorem jsme 100krát vygenerovali posloupnost bitů délky milion a na ně aplikovali uvedené testy. Následující tabulka uvádí, kolikrát daný generátor prošel testem.

	BBS	Geffe	Decim
Frekvenční monobit test	83	26	99
Frekvenční monobit test (mod.)	83	26	99
Modifikovaný Poker test	91	46	20
Runs test	97	47	100

První tři testy jsme uvažovali na hladině významnosti $\alpha = 0,05$. Jako příklad uvádíme v následující tabulce pro každý generátor deset naměřených p-hodnot(náhodně vybraných ze sta, které jsme spočetli) zaokrouhlených na čtyři desetinná místa.

BBS	Geffe	Decim
0.6438	0.0712	0.5726
0.0069	0	0.7905
0.6803	0.133	0.4447
0.7417	0.0381	0.6676
0.1931	0	0.1702
0.0465	0.2602	0.7071
0.3799	0	0.21
0.1076	0.2084	0.4438
0.1096	0	0.8544
0.77	0	0.7385

Monomiální testy

V této části uvedeme výsledky Afinního konstantního testu, d-monomiálního testu a Monomiálního distribučního testu. Protože BBS generátor nepoužívá klíč a ani inicializační vektor, uvedené testy jsme aplikovali jenom na Geffe generátor a na Decim. Pro jednotlivé testy jsme používali následující parametry:

- Afinní test: $n = 1024$, $m = 10$ a $\alpha = 0,05$
- d-monomiální test: $n = 10$ a $\alpha = 0,05$
- Monomiální distribuční test: $P = 10$, $n = 6$ a $\alpha = 0,05$

Pro Geffe generátor i pro Decim jsme aplikovali uvedené testy celkem 100krát a v následující tabulce uvádíme, kolikrát daný generátor prošel testem.

	Geffe	Decim
Afinní konstantní test	93	90
d-monomiální test	0	0
Monomiální distribuční test	0	0

Strukturální analýza

Klíč/Keystream test, IV/Keystream test, Korelační test pro rámec a Difuzní test byly aplikovány na Geffe generátor a Decim. Protože BBS generátor nepoužívá klíč ani inicializační vektor, nebyly tyto testy na BBS aplikovány. Pro jednotlivé testy jsme používali následující parametry:

- Klíč/Keystream test, IV/Keystream test: $m = 2^{20}$
- Korelační test pro rámec, Difuzní test: $m = 2^{20}, l = 192$

Pro Geffe generátor i pro Decim jsme aplikovali uvedené testy celkem 100krát a v následující tabulce uvádíme, kolikrát daný generátor prošel testem.

	Geffe	Decim
Klíč/Keystream test	95	5
IV/Keystream test	95	4
Korelační test pro rámec	0	0
Difuzní test	0	0

Využití Geffe generátoru a Decimu v Pollardově ρ metodě

V této části si uvedeme výsledky využití Geffe generátoru a Decimu v Pollardově ρ metodě pro faktorizaci čísel. Při testování jsme zkoumali počet iterací, které vykonala Pollardova ρ metoda. Náhodně jsme si vygenerovali 100 složených čísel, kterých nejmenší dělitěl byl rovný alespoň 7. Ve většině případů dosáhla nejmenší počet iterací Pollardova ρ metoda s polynomem $f(x) = x^2 + 1 \in \mathbb{Z}_n[x]$ a s inicializačním prvkem $x_0 = 1$. Jako příklad uvádíme v následující tabulce počet iterací pro testované tři případy. Složené číslo v levém sloupci, které jsme faktorizovali, uvádíme v prvočíselném rozkladu.

	$x^2 + 1$	Decim	Geffe
$1112323 \cdot 1238491 \cdot 7534327$	1680	645175	1078921
$2123603 \cdot 3745663 \cdot 4159787$	614	316721	fail
$17 \cdot 1213 \cdot 1523 \cdot 1741254515257$	6	13	32
$11 \cdot 557 \cdot 2267 \cdot 845701061299$	4	2	12
$43261 \cdot 54361 \cdot 71437$	185	30330	10962
$12433 \cdot 18793 \cdot 25261$	45	2656	5627
$565891 \cdot 565891$	523	622098	144163
$23 \cdot 47 \cdot 1009 \cdot 743803$	6	3	2
$101 \cdot 5556367$	9	87	112
$1597 \cdot 15901$	36	740	820

Závěr

V této práci jsme se věnovali několika různým typům testů, pomocí kterých jsme otestovali BBS generátor, Geffe generátor a proudovou šifru Decim. V základních statistických testech jsme pro Decim naměřili zajímavé výsledky. Ikdyž Decim podle našich výsledků uspěl ve Frekvenčním monobit testu, jeho modifikaci i v Runs testu, tak v modifikovaném Poker testu uspěl jenom v 20% případů. U základních testů nejvíckrát uspěl BBS generátor.

Monomiální distribuční test i d-monomiální test se podle našich výsledků pro Geffe generátor a Decim jeví jako velmi účinné. Ve 100% případů tyto testy správně odhalily šifru.

Strukturální analýza odhalila slabiny v inicializační fázi pro Decim i pro Geffe generátor. Oba generátory ani jednou neprošly přes Korelační test pro rámec a ani přes Difuzní test. Geffe generátor v 95% případů prošel Klíč/Keystream testem i IV/Keystream testem. Avšak Decim těmito testy prošel jenom zhruba v 5% případů.

Nakonec jsme otestovali využití Geffe generátoru a Decimu v Pollardově ρ metodě pro faktorizaci složených čísel. Zkoumali jsme počet iterací, které vykonala Pollardova ρ metoda. Podle našich výsledků dosáhla ve většině případů nejmenší počet iterací klasická Pollardova ρ metoda s polynomem $f(x) = x^2 + 1 \in \mathbb{Z}_n[x]$, tj. bez využití Geffeho generátora a Decimu.

Seznam použité literatury

[1] Václav Dupač, Marie Hušková, *Pravděpodobnost a matematická statistika*, Karolinum, 2005

[2] Jiří Anděl, *Základy matematické statistiky*, MatfyzPress, 2007

[3] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996

[4] Vlastimil Klíma, *Základy moderní kryptologie - Symetrická kryptografie II.*, skripta k přednášce na MFF UK Úvod do klasických a moderních metod šifrování, 2006

[5] Rudolf Lidl, Harald Niederreiter, *Introduction to Finite Fields and their Applications*, Cambridge University Press, 1986

[6] Thomas W. Cusick, Pantelimon Stanica, *Cryptographic Boolean Functions and Applications*, Academic Press, 2009

[7] Pascal Junod, *Cryptographic Secure Pseudo-Random Bits Generation: The Blum-Blum-Shub Generator*, studentská práce, ETH Zürich, 1999

[8] Volodymyr Maksymovych, Oleh Harasymchuk, Yuriy Kostiv, *Poisson Pulse Sequence Generators based upon Modified Geffe Generators*, Technical Transaction: Automatic Control, vol. 3-AC/2013, pp. 17-28, 2013

[9] SP 800-22 Rev. 1a. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, Technical Report, NIST, 2010

[10] C. Berbain, O. Billet, A. Canteaut, N. Courtois, B. Debraize, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin, H. Sibert, *Decim, A New Stream Cipher for Hardware Applications*, ECRYPT Stream Cipher Workshop SKEW 2005

[11] eSTREAM projekt, `http://www.ecrypt.eu.org/stream/`

[12] A. Gouget, H. Sibert, C. Berbain, N. Courtois, B. Debraize, C. Mitchell, *Analysis of the Bit-Search Generator and Sequence Compression Techniques*, Fast Software Encryption - FSE 2005, Lecture Notes in Computer Science. Springer-Verlag, 2005

[13] Wael M. F. Abdel-Rehim, Ismail A. Ismail, Ehab Morsy, *Testing randomness: Implementing Poker Approaches with Hands of Four Numbers*, IJCSI, Vol. 9, Issue 4, No 3, 2012

[14] Victor H. Moll, *Numbers and Functions: From a Classical-Experimental Mathematician's Point of View*, American Mathematical Society, 2012

[15] Sheldon M. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, 4. vydání, Academic Press, 2009

[16] Markku-Juhani O. Sarinen, *Chosen-IV Statistical Attacks on eStream Ciphers*, eSTREAM, ECRYPT Stream Cipher Project, Report 2006/013, pp.5-19, 2006

[17] Eric Filiol, *A New Statistical Testing for Symmetric Ciphers and Hash Functions*, ICICS 2002, LNCS 2513, pp. 342-353, 2002

[18] Hakan Englund, Thomas Johansson, Meltem S. Turan, *A Framework for Chosen IV Statistical Analysis of Stream Ciphers*, INDOCRYPT 2007, LNCS 4859, pp. 268-281, 2007

[19] Meltem S. Turan, Ali Doganaksoy, Cagdas Calik, *Detailed Statistical Analysis of Synchronous Stream Ciphers*, Technical Report 2006/043, Institute of Applied Mathematics, Middle East Technical University, 2006

[20] Henri Cohen, *A Course in Computational Algebraic Number Theory*, Graduate Texts in Mathematics 138, Springer-Verlag, 3rd corrected printing, 1996

[21] Kostas Bimpikis, Ragesh Jaiswal, *Modern Factoring Algorithms*, A technical Report Presented to the University of California, pp. 1-15, 2005

[22] Apache Commons Math 3.5, `http://commons.apache.org/proper/commons-math/`

[23] Marc Briceno, Ian Goldberg, David Wagner, *A pedagogical implementation of A5/1*, 1999, `http://www.scard.org/gsm/a51.html`