Kapitola 5

Generátory náhodných a pseudonáhodných čísel, generátory prvočísel

V roce 1917 si Gilbert Vernam nechal patentovat šifru, která nyní nese jeho jméno. Byl přesvědčen, že je to zcela bezpečná šifra, přestože matematický důkaz jeho tvrzení předložil až o 30 let později C.E. Shannon. Vernamova šifra spočívá v tom, že k otevřenému textu přičteme klíč stejné délky, který je náhodný. To znamená, že pokud známe libovolně dlouhou část klíče, pak všechna písmena abecedy, kterou používáme, se na následujícím místě klíče mohou objevit se stejnou pravděpodobností. Tak například v případě mezinárodní abecedy o 26 písmenech je pravděpodobnost výskytu nějakého písmene na daném místě klíče vždy $1/26$ nezávisle na tom, jaká písmena jsou na jiných místech klíče. Používáme-li pouze číslice $0, 1$, pak je pravděpodobnost výskytu každé z těchto dvou cifer na libovolném místě vždy $1/2$, cifry na jiných místech klíče na tuto pravděpodobnost nemají žádný vliv. Pro bezpečnost Vernamovy šifry je důležité, že žádný klíč nepoužijeme dvakrát pro zašifrování dvou různých zpráv. Dvojí použití stejného klíče bezpečnost Vernamovy šifry dramaticky snižuje, jak si za chvíli ukážeme.

Vernamovu šifru je nutné odlišit od *knižní šifry*, používané v minulých stoletích zejména špiony. Pro ně bylo důležité nemít doma žádné šifrovací pomůcky, které by bylo možné odhalit při domovní prohlídce. Jednou možností bylo použít jako zdroj klíče nějakou předem smluvenou knihu. Šifrování pak probíhalo tak, že k otevřenému textu špion přičetl jako klíč část textu příslušné knihy. Poznamenal si, kde klíč skončil, a při šifrování další zprávy použil následující úsek knihy, která začínala prvním dosud nepoužitým písmenem.Důležité bylo nikdy žádné písmeno knihy nevynechat a stejně tak žádné nepoužít dvakrát. Jen tak bylo možné zajistit, aby oprávněný příjemce zprávy použil k dešifrování stejný klíč. Knižní šifra má s Vernamovou šifrou společné to, že používá k šifrování klíč téže délky jako je délka otevřeného textu, liší se ale v tom, že klíč není náhodná posloupnost písmen. Je-li klíčem text v češtině a pokud známe, že část klíče je `STRCPRSTSKR`, pak ne všechna písmena abecedy jsou stejně pravděpodobná jako následující písmeno klíče. Nenáhodnost klíče lze využít při luštění šifrového textu. Odečteme od něho na všech možných místech nějaké často používané krátké slovo v českých textech, například slovo `ALE`. A díváme se, které části by mohly skutečně být posloupnostmi písmen v otevřeném českém textu. Tím získáme hypotézy o kratičkých částech otevřeného textu a ty se můžeme pokusit rozšiřovat pomocí znalostí češtiny.

V počítačové éře kryptografie je pravděpodobnější, že otevřené texty budou napřed zakódovány posloupností 0 a 1, a teprve potom zašifrovány. V takovém případě je obvyklé použít jako klíč rovněž nějakou posloupnost 0 a 1. Tyto dvě posloupnosti pak binárně sečteme a dostaneme šifrový text. Zde si můžeme ukázat, proč je dvojí použití stejného klíče nebezpečné. Pokud použijeme stejnou posloupnost k jako klíč k zašifrování dvou různých otevřených textů p_1 a p_2, dostaneme šifrové texty

$$s_1 = p_1 \oplus k, \quad s_2 = p_1 \oplus k.$$

Pokud má kryptoanalytik podezření, že klíč byl použit dvakrát, sečte prostě oba šifrové texty a dostane

$$s_1 \oplus s_2 = (p_1 \oplus k) \oplus (p_2 \oplus k) = p_1 \oplus p_2.$$

Text, který tak dostane, je součtem dvou otevřených textů. To je totéž, jako kdyby jeden z nich byl zašifrován pomocí knižní šifry. Proto lze takovýto součet dvou otevřených textů rozluštit stejně jako knižní šifru.

Přestože je Vernamova šifra teoreticky zcela bezpečná, nedočkala se širokého využití. Její problém spočívá v tom, že chceme-li zašifrovat posloupnost n bitů, musíme oprávněnému příjemci zprávy poslat bezpečným kanálem náhodný klíč o délce n bitů. Máme-li takto bezpečný kanál k dispozici, můžeme jej použít přímo pro přenos otevřeného textu.

Absolutní bezpečnost Vernamovy šifry nebudeme formálně dokazovat, důkaz pouze naznačíme. Absolutní bezpečnost spočívá v tom, že znalost šifrového textu žádným způsobem nemění pravděpodobnost otevřeného textu. Odečteme-li od šifrového textu libovolný otevřený text, dostaneme jednoznačně určenou hodnotu klíče, pomocí kterého můžeme z tohoto otevřeného textu dostat daný šifrový text. Protože všechny možné hodnoty klíče jsou při použití Vernamovy šifry stejně pravděpodobné, nemůžeme na základě znalosti šifrového textu usoudit vůbec nic o otevřeném textu.

Otázka, jaká posloupnost bitů je náhodná a jaká ne, je otázka spíše filosofická. Za náhodnou posloupnost bitů považujeme například výsledek házení zcela souměrnou mincí: $panna = 0$, $orel = 1$. Přitom výsledek tohoto házení je zcela určený deterministickými zákony klasické mechaniky. Tak kde je jaká náhodnost? Náhodnost je důsledkem toho, výsledek panna nebo orel závisí na počátečních podmínkách (tj. výšce hodu a rotaci, kterou minci udělíme v okamžiku, kdy opouští naši ruku, tj. síle, kterou na minci působíme) tak jemně, že není v možnostech nejmodernější techniky tuto sílu změřit s takovou přesností, aby řešení příslušné soustavy diferenciálních rovnic umožnilo jakkoliv předpovědět výsledek hodu.

Stejně tak bychom mohli tahat ze dvou čísel v klobouku. Zde vstupuje do hry ještě naše vůle, ani v tomto případě ale neexistuje žádná shoda v tom, nakolik je naše vůle skutečně svobodná a nakolik je deterministická v důsledku působení fyzikálních zákonů v našem mozku a během jeho celého vývoje.

Dalším běžně používaným generátorem náhodných čísel je měření časového úseku mezi dvěma úhozy do klávesnice počítače s přesností na tisíciny vteřiny. Pokud je desetinná část této doby lichá, tak jsme generovali 1, pokud je sudá, tak jsme generovali 0. Jiné fyzikální generátory náhodných čísel jsou založené na náhodnosti radioaktivního rozpadu částic, na kosmickém záření, apod.

Takto tedy můžeme generovat posloupnosti náhodných bitů. Z nich už čistě matematickými prostředky můžeme generovat posloupnosti náhodných čísel z množiny $\{0, 1, 2, \ldots, m\}$. Označíme si

$$n = \lfloor \log_2 m \rfloor + 1,$$

kde $\lfloor x \rfloor$ označuje *celou část* reálného čísla x, tj. největší celé číslo menší nebo rovné x. Potom vezmeme nějakou náhodně generovanou posloupnost bitů $b_0, b_1, \ldots, b_{n-1}$ a spočteme číslo

$$a = \sum_{i=0}^{n-1} b_i 2^i.$$

Dostaneme tak nějaké nezáporné celé číslo menší než $2^n > m$. Pokud je větší než m, tak je zahodíme, pokud je menší nebo rovné m, tak je použijeme. Lze potom dokázat, že takto generovaná posloupnost čísel je náhodná posloupnost čísel z množiny $\{0, 1, \ldots, m\}$.

Pokud potřebujeme generovat náhodná čísla, která mají ve dvojkovém vyjádření přesně n bitů, vezmeme náhodně generovanou posloupnost bitů $b_0, b_1, \ldots, b_{n-2}$ a spočítáme číslo

$$a = 2^{n-1} + \sum_{i=0}^{n-2} b_i 2^i.$$

Pro praktické použití Vernamovy šifry potřebujeme nějak zmenšit délku klíče, kterou musíme oprávněnému příjemci zprávy sdělit zcela bezpečným kanálem. K tomu jsou využívány *generátory pseudonáhodných čísel.* Generátor pseudonáhodných čísel je deterministický algoritmus, který ze vstupních dat v podobě nějaké krátké posloupnosti náhodných bitů vyprodukuje dlouhou posloupnost bitů, která *vypadá* náhodně. Tím se myslí to, že neexistuje žádný "rychlý" algoritmus, který by tuto posloupnost dokázal odlišit od skutečně náhodné posloupnosti. "Rychlým" algoritmem se obvykle rozumí algoritmus, který vyžaduje dobu polynomiálně závisející na délce posloupnosti n. Formálně můžeme generátor pseudonáhodné posloupnosti bitů definovat následovně.

Definice 5.1 Generátor pseudonáhodné posloupnosti bitů typu (k, ℓ) *je nějaká funkce*

$$f : \{0,1\}^k \to \{0,1\}^\ell,$$

kde $\ell > k$ je předem daná hodnota polynomiálně závisející na proměnné k, přičemž hodnotu $f(k)$ lze spočítat v polynomiálním čase (v závislosti na k). Vstupní hodnotu $s_0 \in \{0,1\}^k$ budeme nazývat semínko, *anglicky* seed, *a výstupní hodnotu $f(s_0)$ nazýváme* pseudonáhodná posloupnost bitů.

Odesílatel zprávy a oprávněný příjemce se musí předem domluvit na tom, jaký generátor pseudonáhodné posloupnosti bitů použijí, a potom si musí pomocí bezpečného kanálu sdělit pouze semínko. Stejný generátor pak vygeneruje stejnou pseudonáhodnou posloupnost bitů, kterou oba použijí jako klíč, jeden k šifrování a druhý k dešifrování. Semínko tak funguje jako klíč a generátor pseudonáhodné posloupnosti bitů, který ze semínka vytvoří klíč pro jedno použití, pak můžeme považovat ya součást šifrovacího algoritmu.

Nejjednodušší (a jak si za chvilku ukážeme také nepříliš bezpečný) generátor pseudonáhodné posloupnosti bitů je v angličtině nazýván *linear feedback shift register, LFSR*, česky bychom mohli říkat *lineární generátor se zpětnou vazbou.* Ten spočítá v tom, že nějaký člen posloupnosti spočítáme na základě znalosti předchozích m bitů této posloupnosti z lineárního vztahu

$$z_{i+m} = \sum_{j=0}^{m-1} c_j z_{i+j},$$

kde všechny aritmetické operace děláme ve dvouprvkovém tělese, odkud jsou také koeficienty $c_0, c_1, \ldots, c_{m-1}$. Číslo m můžeme nazývat *řád*, případně *stupeň* lineárního generátoru. Semínkem je potom náhodně zvolená posloupnost bitů $z_0, z_1, \ldots, z_{m-1}$. Pomocí uvedeného vzorce pak můžeme dopočítat dalších maximálně $2^m - m - 1$ bitů, než se začne posloupnost opakovat. To proto, že pokud člen posloupnosti závisí na předchozích m prvcích této posloupnosti, může mít posloupnost nejvýše $2^m - 1$ různých členů. V posloupnosti se nemůže vyskytnout m po sobě jdoucích 0, protože potom by celá posloupnost sestávala pouze z prvků 0. A protože semínko má m prvků, můžeme nově spočítat nejvýše $2^m - 1 - m$ bitů, než se začne posloupnost opakovat. Jako příklad můžeme uvést generátor určený *Fibonacciho rovností*

$$z_{i+2} = z_i + z_{i+1}.$$

Je to lineární generátor řádu 2 s koeficienty $c_0 = c_1 = 1$. Pokud zvolíme jako semínko obvyklou posloupnost $0, 1$, Fibonacciho generátor vytvoří posloupnost

$$0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1 \ldots$$

Posloupnost má periodu $3 = 2^2 - 1$, což je maximální perioda pro posloupnost vytvořenou lineárním generátorem řádu 2. Všimněte si, že první tři dvojice v posloupnosti jsou $0, 1$ a $1, 1$ a $1, 0$, což jsou všechny prvky množiny $\{0, 1\}^2$, které se nerovnají nulové posloupnosti.

Lineární generátory řádu 2 nejsou pro kryptografy zajímavé. Co třeba lineární generátory vyššího řádu? Vzorec

$$z_{i+4} = z_i + z_{i+1}$$

definuje lineární generátor řádu 4, jeho perioda tak může být maximálně $2^4 - 1 = 15$. Zvolíme-li semínko $1, 1, 1, 1$, dostaneme posloupnost

$$1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1 \ldots,$$

která má skutečně periodu 15. Kterýkoliv z nenulových vektorů délky 4 tak můžeme použít jako semínko. Podobně si můžete ověřit, že také vztah

$$z_{i+4} = z_i + z_{i+3}.$$

generuje posloupnost s periodou 15.

Pro ty, kteří již znají *Eulerovu funkci* ϕ, která každému kladnému celému číslu n přiřazuje počet $\phi(n)$ přirozených čísel menších než n, která jsou nesoudělná s n, uvádíme tvrzení, které udává počet lineárních generátorů řádu m, které generují posloupnost s maximální periodou $2^m - 1$. Tento počet se rovná

$$\frac{\phi(2^m - 1)}{m}.$$

Pro $m = 4$ je tento počet rovný $\phi(15)/4 = 2$. Dva uvedené vztahy tak určují jediné lineární generátory řádu 4, které generují posloupnost s maximální periodou 15. Pokud potřebujeme generovat posloupnost s periodou $2^{23} - 1$, můžeme spočítat, že existuje

$$\frac{2^{23} - 1}{23} = \frac{47 \cdot 178\,481}{23} = \frac{46 \cdot 178\,480}{23} = 356\,960$$

lineárních generátorů řádu 23, které mají tuto vlastnost.

Lineární generátory se v praxi nepoužívají, protože nejsou bezpečné. Tím se rozumí to, že ze znalosti $2m$ po sobě jdoucích prvků pseudonáhodné posloupnosti bitů vytvořené lineárním generátorem řádu m můžeme spočítat koeficienty $c_0, c_1, \ldots, c_{m-1}$, které lineární generátor určují, a tím spočítat celou pseudonáhodnou posloupnost bitů, kterou tento generátor generuje. Pokud známe řád m lineárního generátoru, potřebujeme spočítat koeficienty $c_0, c_1, \ldots, c_{m-1}$, abychom věděli vše potřebné. Známe-li posloupnost $x_1, x_2, \ldots, x_n$ bitů otevřeného textu a odpovídající posloupnost $y_1, \ldots, y_n$ bitů šifrového textu, dopočítáme příslušnou posloupnost $z_1, \ldots, z_n$ bitů klíče ze vztahu $z_i = x_i + y_i$. Neznámé koeficienty $c_0, c_1, \ldots, c_{m-1}$ spočítáme ze soustavy lineárních rovnic

$$z_{i+m} = \sum_{i=0}^{m-1} c_j z_{i+j}$$

o m neznámých nad dvouprvkovým tělesem. Pokud je $n \geq 2m$, dostáváme soustavu m lineárních rovnic o m neznámých, kterou můžeme vyřešit. Tuto soustavu můžeme zapsat v maticovém tvaru

$$\begin{pmatrix} z_1 & z_2 & \cdots & z_m \\ z_2 & z_3 & \cdots & z_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ z_m & z_{m+1} & \cdots & z_{2m-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{m-1} \end{pmatrix} = \begin{pmatrix} z_{m+1} \\ z_{m+2} \\ \vdots \\ z_{2m} \end{pmatrix}.$$

Matice této soustavy je regulární, pokud je posloupnost $z_1, \ldots, z_{2m}$ skutečně generována lineárním generátorem řádu m. V takovém případě lze snadno dokázat, že sloupce matice jsou lineárně nezávislé.

Lineární kongruenční generátor

Jiným dobře známým generátorem pseudonáhodných posloupností bitů je *lineární kongruenční generátor*. Parametry tohoto generátoru jsou celé číslo $M \geq 2$ a celá čísla a, b, pro která platí $1 \leq a, b \leq M - 1$. Definujeme $k = \lceil \log_2 M \rceil$ (nejmenší celé číslo větší nebo rovné $\log_2 M$) a zvolíme ℓ z intervalu $k + 1 \leq \ell \leq M - 1$. Zvolíme semínko v podobě celého čísla s_0, kde $0 \leq s_0 \leq M - 1$, a definujeme

$$s_i = (as_{i-1} + b) \bmod M$$

pro $1 \leq i \leq \ell$ (v posledním vztahu používáme obvyklé aritmetické operace s celými čísly) a potom definujeme

$$f(s_0) = (z_1, z_2, \ldots, z_\ell),$$

kde

$$z_i = s_i \bmod 2$$

pro $i = 1, 2, \ldots, \ell$. Dostáváme tak (k, ℓ)-lineární kongruenční generátor.

Příklad 5.2 Volbou parametrů $M = 31$, $a = 3$, $b = 5$ dostáváme $k = 5$. Dále zvolíme $\ell = 10$ a $s_0 = 0$. Budeme postupně počítat další čísla ze vztahu $s_i = (as_{i-1} + b) \bmod M$. Dostaneme tak posloupnost

$$0, 5, 20, 3, 14, 16, 22, 9, 1, 8, 29, 30, 2, 11, 7, 26,$$
$$21, 6, 23, 12, 10, 4, 17, 25, 18, 28, 27, 24, 15, 19.$$

Pro libovolnou volbu s_0 z této posloupnosti čísel pak dostaneme pseudonáhodnou posloupnost bitů $z_1, \ldots, z_{10}$ jako posloupnost zbytků při dělení následujících 10 čísel číslem 2. Tak například původní volba $s_0 = 0$ dává pseudonáhodnou posloupnost bitů 1,0,1,0,0,0,1,1,0,1, volba $s_0 = 19$ dává posloupnost 0,1,0,1,0,0,0,1,1,0, atd.

Pokud bychom náhodou zvolili číslo $s_0 = 13$, které se ve výše spočítané posloupnosti nevyskytuje, dostali bychom

$$s_1 = 3 \cdot 13 + 5 = 13 \bmod 31,$$

tj. při volbě $s_0 = 13$ bychom dostali posloupnost 1,1,1,1,1,1,1,1,1,1.

RSA-generátor

Tento generátor je určený dvěma $(k/2)$-bitovými prvočísly p, q a číslem b takovým, že $\gcd(b, (p-1)(q-1)) = 1$. Označíme $n = pq$. Dále zvolíme

ℓ tak, aby platilo $k < \ell \leq pq$, a semínko $s_0 \in \{1, 2, \ldots, n-1\}$, tj. prvek multiplikativní grupy okruhu celých čísel modulo $n = pq$. Číslo s_0 je tak nenulové k-bitové číslo. Definujeme

$$s_{i+1} = s_i^b \bmod n,$$

a dále definujeme

$$f(s_0) = z_1, z_2, \ldots, z_\ell,$$

kde

$$z_i = s_i \bmod 2.$$

Dostáváme tak (k, ℓ)-RSA generátor.

Příklad 5.3 Zvolíme $n = 263 \cdot 347$, $b = 1547$ a $s_0 = 75364$. Prvních 14 bitů generovaným RSA-generátorem určeným těmito parametry najdete v posledním sloupci následující tabulky

i	s_i	z_i
0	75634	
1	31483	1
2	31238	0
3	51968	0
4	39796	0
5	28716	0
6	14089	1
7	5923	1
8	14089	1
9	62284	0
10	11889	1
11	43467	1
12	71215	1
13	10401	1
14	77444	0

Ze semínka $s_0 = 75634$ tak dostáváme následující pseudonáhodnou posloupnost bitů 1,0,0,0,0,1,1,1,0,1,1,1,1,0.

RSA-generátor je kryptograficky bezpečný v tom smyslu, že nelze předpovědět následující bit posloupnosti s pravděpodobností významně větší než 1/2 za předpokladu, že neexistuje polynomiální algoritmus pro faktorizaci celých čísel.

Nerozlišitelná rozdělení pravděpodobnosti

Generátory pseudonáhodných posloupností bitů by měly splňovat dva požadavky. Měly by být rychlé a měly by být bezpečné. Tyto dva požadavky jsou často v rozporu. Lineární generátor a lineární kongruenční generátor jsou skutečně velmi rychlé, nejsou ale příliš bezpečné. Pro kryptografické použití se nehodí.

Pokusíme se nyní přesněji formulovat, co myslíme tím, že generátor pseudonáhodné posloupnosti bitů je "bezpečný". Intuitivně to znamená, že posloupnost ℓ bitů generovanou tímto generátorem, by nemělo být možné v polynomiálním čase (v závislosti na k nebo, což je totéž, v závislosti na ℓ) rozlišit od skutečně náhodné posloupnosti ℓ bitů. Odtud vyplývá následující definice rozlišitelnosti dvou rozdělení pravděpodobnosti.

Definice 5.4 *Předpokládáme, že p_0 a p_1 jsou dvě rozdělení pravděpodobnosti na množině $\{0,1\}^\ell$ všech posloupností bitů délky ℓ. Dále předpokládáme, že $\mathbf{A} : \{0,1\}^\ell \to \{0,1\}$ je nějaký pravděpodobnostní algoritmus, který funguje v polynomiálním čase (v závislosti na ℓ), a že $\epsilon > 0$. Pro $j = 0,1$ definujeme*

$$E_{\mathbf{A}}(p_j) = \sum_{(z_1,\ldots,z_\ell)\in\{0,1\}^\ell} p_j(z_1,\ldots,z_\ell)\cdot p(\mathbf{A}(z_1,\ldots,z_\ell) = 1|(z_1,\ldots,z_\ell)).$$

Říkáme, že $\mathbf{A}$ ϵ-rozlišuje *p_0 a p_1, pokud*

$$|E_{\mathbf{A}}(p_0) - E_{\mathbf{A}}(p_1)| \geq \epsilon,$$

a říkáme rovněž, že p_0 a p_1 jsou ϵ-rozlišitelné, *pokud existuje algoritmus $\mathbf{A}$, který ϵ-rozlišuje p_0 a p_1.*

Tato definice vyžaduje řadu vysvětlení. Výraz

$$p(\mathbf{A}(z_1,\ldots,z_\ell) = 1|(z_1,\ldots,z_\ell))$$

označuje *podmíněnou pravděpodobnost.* Jsou-li $\mathbf{X}$ a $\mathbf{Y}$ dvě náhodné proměnné, pak podmíněná pravděpodobnost $p(x|y)$ označuje pravděpodobnost, že náhodná proměnná $\mathbf{X}$ nabývá hodnotu x víme-li, že náhodná proměnná $\mathbf{Y}$ nabývá hodnotu y. Je-li $p(x)$ pravděpodobnost, že $\mathbf{X}$ nabývá hodnotu x, dále $p(y)$ je pravděpodobnost, že $\mathbf{Y}$ nabývá hodnotu y a $p(x,y)$ je pravděpodobnost, že současně $\mathbf{X}$ nabývá hodnotu x a $\mathbf{Y}$ nabývá hodnotu y, pak podmíněnou pravděpodobnost $p(x|y)$ můžeme spočítat ze vztahu

$$p(x,y) = p(x|y)p(y).$$

Algoritmus **A** se pokouší rozhodnout, jestli posloupnost bitů $(z_1, \ldots, z_\ell)$ vznikla z rozdělení pravděpodobnosti p_1 nebo z rozdělení pravděpodobnosti p_0. Tento algoritmus může používat náhodná čísla, tj. může to být pravděpodobnostní algoritmus. Výstup $\mathbf{A}(z_1, \ldots, z_\ell)$ tak udává odhad algoritmu, které ze dvou rozdělení pravděpodobnosti p_0 a p_1 vyprodukovalo posloupnost $(z_1, \ldots, z_\ell)$. Číslo $E_{\mathbf{A}}(p_i)$ pak udává průměrnou (očekávanou) hodnotu výstupu algoritmu **A** pro rozdělení pravděpodobnosti p_i. Ta se spočítá tak, že se sečtou přes všechny posloupnosti $(z_1, \ldots, z_\ell) \in \{0,1\}^\ell$ součiny pravděpodobnosti ℓ-tice $(z_1, \ldots, z_\ell)$ a pravděpodobnosti, že algoritmus **A** dá výstup 1, pokud dostane vstup $(z_1, \ldots, z_\ell)$. Algoritmus **A** ϵ-rozlišuje p_0 a p_1, pokud se tyto dvě očekávané hodnoty liší aspoň o ϵ.

Pokud je posloupnost $(z_1, \ldots, z_\ell)$ náhodná, je každý bit zvolen náhodně s pravděpodobností $1/2$ a nezávisle na ostatních bitech posloupnosti, každou posloupnpost tak dostaneme se stejnou pravděpodobností $1/2^\ell$. Toto uniformní rozdělení pravděpodobnosti označíme p_0.

Předpokládejme nyní, že je dán (k, ℓ)-generátor pseudonáhodné posloupnosti bitů, a že jsme k-bitové semínko zvolili náhodně. Dostaneme tak rozdělení pravděpodobnosti na posloupnostech bitů délky ℓ, které si označíme p_1. Za účelem ilustrace ϵ-rozlišitelnosti v následujícím příkladě uděláme zjednodušující předpoklad, že různá semínka vedou k různým pseudonáhodným posloupnostem délky ℓ. Generátor tak z 2^ℓ možných posloupností ℓ bitů generuje pouze 2^k posloupností, každou z nich s pravděpodobností $1/2^k$. Zbývajících $2^\ell - 2^k$ posloupností se tak vůbec nemůže objevit, generátor je produkuje s pravděpodobností 0. Rozdělení p_1 je tak velmi neuniformní.

Přestože jsou tato dvě rozdělení pravděpodobností p_0 a p_1 hodně odlišná, je stále ještě možné, že jsou ϵ-rozlišitelná pouze pro malé hodnoty ϵ. To je také hlavním cílem při konstrukci generátorů pseudonáhodných posloupností bitů.

Příklad 5.5 Předpokládejme, že náš generátor pseudonáhodných posloupností bitů generuje pouze posloupnosti, které mají přesně $\ell/2$ prvků 0 a $\ell/2$ prvků 1. Definujeme (deterministický) algoritmus **A** předpisem

$$\begin{aligned}\mathbf{A}(z_1, \ldots, z_\ell) &= 1, \text{ pokud posloupnost } (z_1, \ldots, z_\ell) \text{ obsahuje } \ell/2 \\ & \text{ bitů rovných } 0,\end{aligned}$$

a

$$\mathbf{A}(z_1, \ldots, z_\ell) = 0, \text{ v opačném případě.}$$

Potom platí

$$E_{\mathbf{A}}(p_0) = \frac{\binom{\ell}{\ell/2}}{2^\ell},$$

zatímco

$$E_{\mathbf{A}}(p_1) = 1.$$

Protože

$$\lim_{\ell \to \infty} \frac{\binom{\ell}{\ell/2}}{2^\ell} = 0,$$

jsou obě rozdělení ϵ-rozlišitelná pro libovolné $\epsilon < 1$, pokud je ℓ dostatečně velké. Takový generátor proto není příliš vhodný.

Předpovídání následujícího bitu

Posloupnost bitů generovaná nějakým generátorem pseudonáhodných čísel nepřipomíná náhodnou posloupnost, pokud můžeme předpovědět i-tý bit posloupnosti na základě znalosti předchozích $i-1$ bitů s pravděpodobností významně větší než $1/2$. Je-li f nějaký generátor pseudonáhodné posloupnosti bitů typu (k, ℓ), číslo $i = 0, 1, \ldots, \ell - 1$, a $\mathbf{B}_i$ je pravděpodobnostní algoritmus, který ze znalosti (vstupu) prvních $i-1$ bitů $z_1, \ldots, z_{i-1}$ bitů posloupnosti generované generátorem f správně předpoví následující bit z_i s pravděpodobností aspoň $1/2 + \epsilon$ pro nějaké $\epsilon > 0$, pak říkáme, že $\mathbf{B}_i$ ϵ-předpovídá následující bit.

Na první pohled se může zdát, že ϵ-předpovědět následující bit je něco jiného, než ϵ-rozlišit posloupnost bitů generovanou generátorem f od náhodné posloupnosti bitů. Ve skutečnosti můžeme z algoritmu $\mathbf{B}_i$ pro ϵ-předpovídání následujícího bitu sestrojit algoritmus $\mathbf{A}$, který ϵ-rozlišuje pseudonáhodnou posloupnost od posloupnosti náhodné. Bez důkazu si ukážeme jak:

vstup: posloupnost $(z_1, z_2, \ldots, z_\ell)$

1. spočítáme $z := \mathbf{B}_i(z_1, \ldots, z_i - 1)$
2. **if** $z = z_i$ **then** $\mathbf{A}(z_1, \ldots, z_\ell) = 1$ **else** $\mathbf{A}(z_1, \ldots, z_\ell) = 0$.

Platí také obráceně, že pokud existuje nějaký algoritmus, který ϵ-rozlišuje rozdělení pravděpodobnosti p_1 určené generátorem f pseudonáhodných čísel typu (k, ℓ) a rovnoměrné rozdělení pravděpodobnosti p_0 na množině $(\mathbf{Z}_2)^\ell$ všech posloupností délky ℓ tvořených čísly $0, 1$, pak pro nějaké $i = 1, \ldots, \ell - 1$ existuje algoritmus $\mathbf{B}_i$, který (ϵ/ℓ)-předpovídá následující bit.

Pro každé $i = 0, 1, \ldots, \ell$ definujeme pravděpodobnostní rozdělení q_i na množině $\mathbf{Z}_2^\ell$ tak, že prvních i bitů je generováno generátorem f a zbývajících

$\ell - i$ bitů je generováno náhodně. Při této definici platí $q_0 = p_0$ a $q_\ell = p_1$. Předpokládáme, že algorimus **A** ϵ-rozlišuje p_0 a p_1, tj.

$$|E_{\mathbf{A}}(q_0) - E_{\mathbf{A}}(q_\ell)| \geq \epsilon.$$

Z trojúhelníkové nerovnosti vyplývá

$$|E_{\mathbf{A}}(q_0) - E_{\mathbf{A}}(q_\ell)| \leq \sum_{i=1}^{\ell} |E_{\mathbf{A}}(q_{i-1}) - E_{\mathbf{A}}(q_i)|,$$

a tak pro nějaké $i = 1, \ldots, \ell$ musí platit

$$|E_{\mathbf{A}}(q_{i-1}) - E_{\mathbf{A}}(q_i)| \geq \frac{\epsilon}{\ell}.$$

Jestliže nějaké takové i známe, pak algoritmus $\mathbf{B}_i$ pro předpovídání následujícího bitu lze na základě algoritmu **A** sestrojit následovně:

vstup: posloupnost $(z_1, z_2, \ldots, z_{i-1})$

1. zvolíme náhodně posloupnost $(z_i, \ldots, z_\ell)$
2. spočítáme $z := \mathbf{A}(z_1, \ldots, z_\ell)$
3. definujeme $\mathbf{B}_i(z_1, \ldots, z_{i-1}) = (z + z_i) \bmod 2$.

Odůvodnění tohoto algoritmu je následující. Rozdělení pravděpodobností q_{i-1} a q_i se liší pouze v i-tém bitu posloupnosti $(z_1, z_2, \ldots, z_\ell)$, v rozdělení q_i je bit z_i produkován generátorem pseudonáhodných čísel f, zatímco v rozdělení q_{i-1} je náhodný. Pokud algoritmus **A** vydá hodnotu $\mathbf{A}(z_1, z_2, \ldots, z_\ell) = 0$, tak si myslí, že posloupnost $(z_1, z_2, \ldots, z_\ell)$ pochází pravděpodobněji z rozdělení q_i, tj. že bit z_i byl generován generátorem f. V tomto případě vezmeme hodnotu $z = z_i$ jako naši předpověď i-tého bitu. Pokud **A** odpoví $\mathbf{A}(z_1, z_2, \ldots, z_\ell) = 1$, pak si myslí, že i-tý bit z_i byl generován náhodně. V takovém případě bereme hodnotu $1 - z_i$ jako předpověď i-tého bitu.

Generátory náhodných prvočísel

V následujících dvou přednáškách o asymetrické kryprografii uvidíme, jak je důležité umět generovat dostatečně velká náhodná prvočísla. V této části přednášky si ukážeme dva testy, pomocí kterých můžeme rozhodnout, je-li nějaké kladné celé číslo prvočíslem. Testování prvočíselnosti je velmi aktuální oblastí výzkumu v matematice. V roce 2002 vyvolal v celé matematické komunitě velký zájem výsledek tří indických matematiků, kteří nalezli první algoritmus, který v polynomiálním čase (v závislosti na n) dokáže rozhodnout, je-li n prvočíslo. My si ukážeme mnohem jednodušší algoritmy, které rozhodují s velkou pravděpodobností, je-li n prvočíslo. Začneme zcela jednoduchým tvrzením.

Věta 5.6 *Je-li n složené číslo, pak existuje prvočíselný dělitel p čísla n, který je nejvýše rovný $\sqrt{n}$.*

Generátory náhodných k-bitových prvočísel začínají tím, že napřed generují náhodné číslo, které má ve dvojkovém vyjádření délku k-bitů. Už jsme si ukázali jak. Potom zkusmo dělí toto číslo všemi prvočísly menšími než nějaká předem daná mez B, typicky se volí $B = 10^6$. Tato prvočísla jsou uchovávána v databázi. Pokud číslo n projde tímto testem zkusmého dělení, použijí se další testy.

Jednoduchý test je založený na *malé Fermatově větě.*

Věta 5.7 *Je-li n prvočíslo, pak*

$$a^{n-1} = 1 \bmod n$$

pro všechna celá čísla a taková, že $\gcd(a, n) = 1$.

Malou Fermatovu větu můžeme použít k důkazu, že nějaké číslo n není prvočíslem. Zvolíme číslo $a \in \{1, 2, \dots, n-1\}$ a potom spočítáme číslo $y = a^{n-1} \bmod n$. Je-li $y \neq 1$, pak je n složené číslo. Pokud je $y = 1$, tak nevíme, je-li nebo není-li n prvočíslem. Je důležité si uvědomit, že pokud Fermatův test ukáže, že n je složené číslo, tak přesto nenajde žádného dělitele n. Fermatův test pouze ukáže, že n nemá vlastnost, kterou má každé prvočíslo. Nevýhodou Fermatova testu je skutečnost, že existují složená čísla, která všem možným Fermatovým testům vyhovují. Takovým číslům se říká *Carmichaelova čísla.* Nejmenší z nich je $561 = 3 \cdot 11 \cdot 17$.

Existuje jiný elementární test, který dokáže o každém složeném čísle rozhodnout, že je složené. Tento test se nazývá *Millerův-Rabinův* test. Ten je založený na následující modifikaci malé Fermatovy věty. Je-li n liché kladné celé číslo, pak označíme

$$s = \max\{r \in \mathbf{N} : 2^r | (n-1)\}.$$

Zde $\mathbf{N}$ označuje množinu všech přirozených čísel. Potom

$$d = \frac{n-1}{2^s}$$

je liché číslo, je to největší lichý dělitel čísla $n-1$.

Věta 5.8 *Je-li n prvočíslo a je-li a celé číslo nesoudělné s n, pak buď*

$$a^d = 1 \bmod n,$$

nebo existuje $r \in \{0, 1, \ldots, s-1\}$, pro které platí

$$a^{2^r d} = -1 \bmod n.$$

Je-li n prvočíslo, pak nastává aspoň jedna z možností uvedených v předchozí větě. Pokud tedy najdeme nějaké číslo a, které je nesoudělné s n a současně nesplňuje ani jednu ze dvou alternativ uvedených v této větě, pak je n složené číslo. O takovém čísle a říkáme, že *dosvědčuje* složenost čísla n.

Příklad 5.9 Vyzkoušíme Millerův-Rabinův test na nejmenší Carmichaelovo číslo $n = 561$. Víme už, že Fermatův test nestačí k důkazu, že 561 není prvočíslo. Ale $a = 2$ dosvědčuje, že číslo $n = 561$ je složené. Dostáváme $s = 4$, $d = 35$, $2^{35} = 263 \bmod 561$, $2^{2 \cdot 35} = 166 \bmod 561$, $2^{4 \cdot 35} = 67 \bmod 561$ a $2^{8 \cdot 35} = 1 \bmod 561$. Z předchozí věty tak plyne, že 561 není prvočíslo.

Pro efektivitu použití Millerova-Rabinova testu je důležité vědět, že existuje dostatečně mnoho čísel, která dosvědčují složenost nějakého složeného čísla n. To ukazuje následující věta.

Věta 5.10 *Je-li $n \geq 3$ liché složené číslo, pak množina $\{1, 2, \ldots, n-1\}$ obsahuje nejvýše $(n-1)/4$ čísel, která jsou nesoudělná s n a nedosvědčují složenost čísla n.*

Chceme-li použít Millerův-Rabinův test na liché kladné celé číslo n, zvolíme náhodné číslo $a \in \{1, 2, \ldots, n-1\}$ a spočítáme společného dělitele $\gcd(a, n)$ čísel a, n. Je-li $\gcd(a, n) > 1$, tak je n složené číslo. V opačném případě spočítáme posloupnost čísel

$$a^d, a^{2d}, a^{4d}, \ldots, a^{2^{s-1}d}.$$

Pokud najdeme číslo, které dosvědčuje složenost n, tak jsme dokázali, že n není prvočíslo. Podle předchozí věty je pravděpodobnost, že n je složené číslo a my jsme nezvolili číslo a, které to dosvědčuje, menší než $1/4$. Opakujeme-li Rabinův-Millerův test t-krát, pak pravděpodobnost, že nezvolíme žádné číslo, které složenost n dosvědčuje, je menší než $(1/4)^t$. Pro $t = 10$ je tato pravděpodobnost nejvýše rovná $1/2^{20} \sim 1/10^6$. Toto číslo je pouze hrubým horním odhadem pravděpodobnosti, že t-násobné použití Rabinova-Millerova testu neodhalí složenost čísla n. Podrobnější analýza ukazuje, že tato pravděpodobnost je pro velká čísla n ve skutečnosti řádově mnohem menší.

Vrátíme-li se ke generátorům náhodných k-bitových prvočísel, pak po zkusmém dělení pokračujeme Rabinovým-Millerovým testem. Pokud volíme $k \geq 1000$, pak už pro $t = 3$ je pravděpodobnost, že n je složené číslo a Rabinův-Millerův test to neodhalí, menší než $(1/2)^{80}$. V takovém případě považujeme n za prvočíslo.