ÚSTAV INFORMAČNÍCH STUDIÍ A KNIHOVNICTVÍ
FF UK V PRAZE

Jiří Ivánek

Vybrané kapitoly z kódování informací

Verze 1.0

Praha

Říjen 2007

Obsah

Předmluva

Do učebního textu pro předmět Kódování informací jsou zahrnuty vybrané metody kódování a šifrování, na nichž je demonstrována podstata řešení problémů rychlého a zabezpečeného přenosu zpráv. Důraz byl kladen na co nejjednodušší výklad, který by umožnil samostatné zvládnutí základních algoritmů a jejich aplikaci na malých příkladech. V nezbytné míře jsou uvedeny též potřebné matematické pojmy a výsledky. Postupy jsou demonstrovány na ukázkových příkladech. Na závěr každé části jsou připojeny příklady k procvičování. Části jsou relativně samostatné, tudíž je lze studovat v libovolném pořadí. K podrobnějšímu studiu problematiky kódování informací lze doporučit monografii

Jiroušek, R. - Ivánek, J. - Máša, P. - Toušek, J. - Vaněk, N.: Principy digitální komunikace. LEDA, Praha 2006.

Kapitola 1

Základy kódování a teorie informace

Předmětem klasické teorie kódování jsou zobrazení konečných množin objektů libovolného druhu nejčastěji do množin řetězců symbolů některé abecedy. Hlavní aplikace jsou v oblasti přenosů informací. Kódovanými objekty jsou v tomto případě jednotlivé zprávy z nějakého informačního zdroje. Zakódovaná zpráva je přenášena informačním kanálem a poté opět dekódována.

Typickou úlohou teorie kódování je konstrukce optimálního kódu. Kritérium optimality je přitom spojeno s minimalizací délky kódu pro kódované objekty. Podle potřeby se ještě kladou různé požadavky na přípustnost kódu, např. existence jednoznačného dekódování, možnost detekce a opravy chyb při přenosu, jednoduchost procesu kódování, apod.

Historicky se kódy objevily v podobě kryptogramů již ve starověku. Teprve moderní doba vytýčila cíl zajistit pomocí kódování rychlou a účelnou přepravu informací – příkladem je Morseův telegrafní kód (Morseova abeceda). Ucelenou teorii komunikace (teorii informace) vybudoval koncem čtyřicátých let minulého století C. Shannon.

V našem výkladu, který vychází z učebnic a monografií [1, 11, 4, 13, 20] se soustředíme na formulaci a řešení základní úlohy konstrukce optimálního kódu. Popíšeme dva algoritmy pro konstrukci efektivních kódů a ukážeme jejich vlastnosti. Zmíníme se rovněž o kódech odhalujících a odstraňujících chyby při přenosu. Na závěr ukážeme základní výsledky Shannonovy teorie týkající se míry informace, kapacity informačních kanálů a existence efektivních kódů.

1.1 Vlastnosti kódů

DEFINICE 1.1 *Nechť A je abeceda symbolů o r prvcích.* r-prvkové kódování *množiny X je každé zobrazení K, které přiřazuje prvkům množiny X konečné řetězce symbolů z A (jednotlivé kódy).*

Jako příkladu si všimněme Morseova kódování latinské abecedy. Zde $X = \{A, B, C, D, \ldots\}$. Budeme-li „tečku“ chápat jako `0` a „čárku“ jako `1`, můžeme Morseovo kódování zapsat takto:

$$K(A) = \texttt{01},\ K(B) = \texttt{1000}, \ldots$$

Ve skutečnosti je ovšem Morseovo kódování trojprvkové – na konci každého kódového ekvivalentu písmene se musí vložit mezera, která je třetím symbolem. Bez použití mezer by zprávy v Morseově abecedě nebylo možno jednoznačně dekódovat. Například `SOS` bychom bez mezer mohli číst třeba jako `VTB`, `EUTNI`, `IJS`, apod.

V dalším se budeme zabývat zejména (dvojprvkovým) kódováním. Většinu poznatků však lze snadno zobecnit na r-prvkové kódování.

Jednoznačnost dekódování je základní požadavek, který na kódy klademe. Z hlediska definice kódování jako zobrazení ovšem nestačí, aby toto zobrazení bylo prosté (tuto vlastnost např. Morseovo kódování má). Potřebná vlastnost kódu se nazývá separabilita.

DEFINICE 1.2 *Kódování K je* separabilní *(jednoznačně dekódovatelné), jestliže pro všechna*
$y_1, \ldots, y_m, z_1, \ldots, z_n \in X$ *platí:*

Jestliže kódy $K(y_1), \ldots, K(y_m)$ zapsané za sebe vytváří stejný řetězec jako $K(z_1), \ldots, K(z_n)$, pak $m = n$ a $y_i = z_i$ pro všechna i.

Separabilní kódování tedy přiřazuje objektům z X po dvou různé kódy a zřetězení kódů přiřazených různým posloupnostem objektů z X jsou navzájem různá.

V dalším se soustředíme na kódování do abecedy $\{0, 1\}$. Abychom se vyhnuli triviálním neužitečným případům, budeme předpokládat, že kódování je prosté a nedává nikdy prázdný řetězec.

Nejdůležitějším případem separabilních kódování jsou takové, v nichž žádný kód není počátkem jiného kódu. Taková kódování jsou rovněž výsledkem algoritmů pro konstrukci efektivních kódů, které budeme později popisovat.

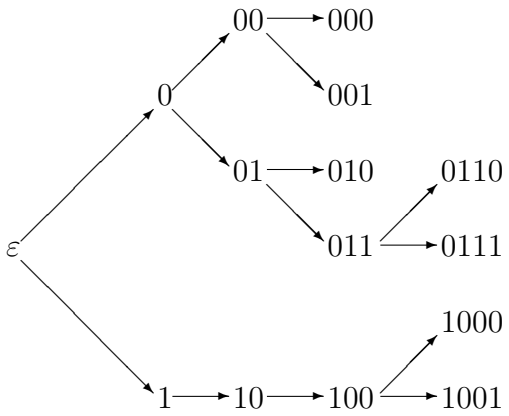
DEFINICE 1.3 *Kódování K má* vlastnost prefixu, *jestliže žádný kód $K(x)$ není prodloužením jiného kódu $K(y)$.*

VĚTA 1.1 *(o kódování s vlastností prefixu)*
Každé kódování, které má vlastnost prefixu, je separabilní.

Příklad 1.1 *Snadno se přesvědčíme, že následující kódování množiny* $X = \{x_0, x_1, \ldots, x_6\}$ *má vlastnost prefixu:*

$x_0 \ldots$ `000`
$x_1 \ldots$ `001`
$x_2 \ldots$ `010`
$x_3 \ldots$ `0110`
$x_4 \ldots$ `0111`
$x_5 \ldots$ `1000`
$x_6 \ldots$ `1001`

Kódování s vlastností prefixu lze přehledně znázornit ve stromu, jehož uzly jsou všechny počáteční úseky kódových slov a hrany vyjadřují připojení `0` nebo `1`. Následující obr. 1.1 zachycuje strom kódu z předchozího příkladu.

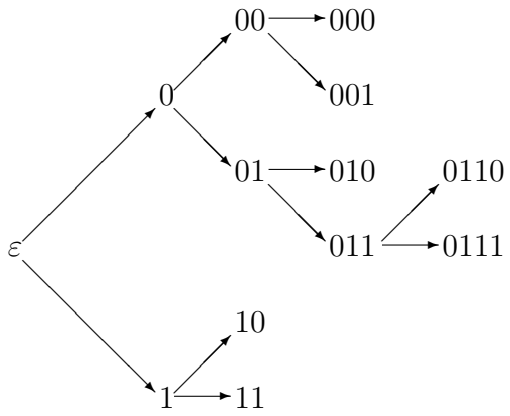


Obrázek 1.1: Strom kódu

Kořenem stromu je prázdný řetězec, listy stromu jsou jednotlivé kódy. Z některých uzlů vystupuje pouze jedna hrana. Znamená to, že například slovo `11` není počátkem žádného kódu – existují posloupnosti `0` a `1`, které "nic nemohou znamenat". Jestliže libovolný řetězec symbolů abecedy je počátkem nějakého kódu, pak kódování v jistém smyslu pokrývá množinu všech řetězců symbolů z abecedy. Tato vlastnost kódování se v souvislosti s optimalizací ukáže důležitou, a proto ji budeme definovat.

DEFINICE 1.4 *Separabilní kódování K je* úplné, *jestliže ke každému řetězci w symbolů kódové abecedy A buď existuje $x_1 \in X$ tak, že w je počátkem kódu $K(x_1)$, nebo existuje $x_2 \in X$ tak, že kód $K(x_2)$ je počátkem řetězce w.*

Kódování s vlastností prefixu je *úplné*, jestliže v jeho kódovém stromě vychází z každého vnitřního uzlu právě dvě hrany. Úplný kód bychom dostali ve výše uvedeném příkladě třeba tak, že bychom "zkrátili" dolní větev stromu takto (viz obr. 1.2).



Obrázek 1.2: Strom úplného kódu

Zajímavá je souvislost různých vlastností kódování s rozložením délek kódů. Nejjednodušší je situace, kdy mají všechny kódy w stejnou délku $d(w)$.

DEFINICE 1.5 *Kódování K množiny X je* stejnoměrné, *jestliže K je prosté zobrazení a pro všechna $x, y \in X$ je $d(K(x)) = d(K(y)) = d$*
(číslo d nazýváme délkou stejnoměrného kódování K).

Zatím jsme uváděli příklady nestejnoměrného kódování. Stejnoměrné kódování má řadu předností (zejména co se týče spolehlivosti přenosu), a proto je často užíváno. Příkladem je třeba kódování alfanumerických znaků a povelů v počítačích.

VĚTA 1.2 *(o stejnoměrném kódování)*
Jestliže množina X má m prvků, pak pro každé $d \geq \log_2 m$ existuje stejnoměrné kódování množiny X s délkou kódování d. Každé takové kódování má vlastnost prefixu; úplné je právě tehdy, když $d = \log_2 m$.

Složitější je situace, když kódy nemají stejnou délku – kódování je nestejnoměrné. Platí následující tři věty (předchozí věta o stejnoměrném kódování je vlastně jejich důsledkem pro případ stejných délek kódů).

VĚTA 1.3 *(Kraft – McMillanova nerovnost)*

1. *Jestliže kódování K množiny X je separabilní, potom*

$$\sum_{x \in X} \frac{1}{2^{d(K(x))}} \leq 1$$

2. *Jestliže $X = \{x_1, \ldots, x_m\}$ a jsou dána přirozená čísla $d_1, \ldots, d_m$ taková, že*

$$\sum_{i=1}^{m} \frac{1}{2^{d_i}} \leq 1$$

pak existuje separabilní kódování K množiny X s kódy daných délek:

$$d(K(x_1)) = d_1, \ldots, d(K(x_m)) = d_m.$$

VĚTA 1.4 *(existence kódování s vlastností prefixu)*
Ke každému separabilnímu kódování $K(X)$ existuje kódování $K'(X)$ s vlastností prefixu takové, že obě mají stejné rozložení délek kódů, tj. pro všechna $x \in X$ je $d(K'(x)) = d(K(x))$.

VĚTA 1.5 *(o úplných kódováních)*
Separabilní kódování K je úplné právě tehdy, když má vlastnost prefixu a platí:

$$\sum_{x \in X} \frac{1}{2^{d(K(x))}} = 1.$$

1.2 Efektivní kódování

V tomto článku se budeme zabývat konstrukcí efektivních kódování za předpokladu, že je známo statistické rozložení četnosti objektů kódované množiny X (obecněji rozložení pravděpodobnosti na X). Hlavní interpretace problému efektivního kódování je v oblasti přenosu zpráv. Zkoumá se informační zdroj, který produkuje posloupnost zpráv (z množiny X). Předpokládáme, že zprávy jsou produkovány na sobě nezávisle. Informační kanál umožňuje přenášet signály `0`,`1` (jeden za jednotku času). Jde o to nalézt kódování zpráv takové, aby průměrná rychlost přenosu zpráv informačním kanálem byla co největší, tj. průměrný počet signálů použitých na zakódování zprávy byl co nejmenší.

DEFINICE 1.6 *Nechť $X = \{x_1, \ldots, x_m\}$ a $P = (p_1, \ldots, p_m)$ je rozdělení pravděpodobnosti na X (tj. $p_i \geq 0, \sum_{i=1}^{m} p_i = 1$). Každému kódování K množiny X s rozdělením pravděpodobností P přiřadíme* střední délku kódu

$$d_p(K) = \sum_{i=1}^{m} p_i \cdot d(K(x_i)).$$

Kódování K_0 množiny X nazveme optimální *při rozdělení pravděpodobnosti P, jestliže K_0 má vlastnost prefixu a pro všechna kódování K množiny X s vlastností prefixu platí:*

$$d_p(K_0) \leq d_p(K).$$

Optimální kódování tedy vybíráme ze třídy všech kódování s vlastností prefixu tak, aby střední délka kódu byla minimální. Všimněme si, že pro stejnoměrné kódování S množiny $X = \{x_1, \ldots, x_m\}$ s délkou slov d platí při libovolném rozdělení pravděpodobnosti:

$$d_p(S) = \sum_{i=1}^{m} p_i \cdot d(K(x_i)) = \sum_{i=1}^{m} p_i \cdot d = d$$

Střední délka stejnoměrného kódování tedy nezávisí na rozdělení pravděpodobnosti na kódované množině. Jestliže rozdělení pravděpodobnosti P je rovnoměrné, tj. $p_i = \frac{1}{m}$ pro $i = 1, \ldots, m$ a $m = 2^d$, bude stejnoměrné kódování optimální. Při nerovnoměrném rozložení pravděpodobností bude optimální kódování vesměs nestejnoměrné – kratší kódy budou přiřazeny objektům s větší pravděpodobností výskytu a naopak delší kódy objektům s menší pravděpodobností výskytu.

Dříve než se seznámíme s algoritmickým řešením problému optimálního kódování, vyslovíme věty o existenci a vlastnostech optimálního kódování.

VĚTA 1.6 *(o existenci optimálního kódování)*
Při libovolném rozdělení pravděpodobností $P = (p_1, \ldots, p_m)$ na množině $X = \{x_1, \ldots, x_m\}$ existuje optimální kódování K množiny X, dokonce takové, že je úplné. (Jestliže jsou všechny pravděpodobnosti p_i nenulové, pak je každé optimální kódování úplné.)

Podle věty 1.6 se při hledání optimálního kódování stačí omezit na úplná kódování s vlastností prefixu. Vzhledem k větám 1.4 a 1.5 je problém konstrukce optimálního kódování ekvivalentní následujícímu optimalizačnímu problému: Nalézt kladná přirozená čísla $d_1, \ldots, d_m$ minimalizující hodnotu

$$\sum_{i=1}^{m} p_i \cdot d_i$$

za podmínky

$$\sum_{i=1}^{m} \frac{1}{2^{d_i}} = 1.$$

Pro optimální střední délku kódu platí odhady nalezené C. Shannonem:

VĚTA 1.7 *(o optimální střední délce kódu)*
Nechť K_0 je optimální kódování na množině $X = \{x_1, \ldots, x_m\}$ při rozdělení pravděpodobnosti $P = (p_1, \ldots, p_m)$. Označme

$$H(X, P) = -\sum_{i=1}^{m} p_i \cdot \log_2 p_i$$

($0 \cdot \log_2 0$ definujeme jako 0). Pak

$$H(X, P) \leq d_p(K_0) \leq H(X, P) + 1,$$

přičemž

$$d_p(K_0) = H(X, P)$$

právě tehdy, když existují přirozená čísla $d_1, \ldots, d_m$ taková, že $p_i = \frac{1}{2^{d_i}}$ pro $i = 1, \ldots, m$.

Poznamenejme, že číslu $H(X, P) = -\sum_{i=1}^{m} p_i \cdot \log_2 p_i$ se říká entropie (podrobněji se jí budeme zabývat v části 1.4), optimální kódování se proto vzhledem větě 1.7 někdy nazývá entropické.

Nejznámější algoritmy pro efektivní kódování předložili postupně C. Shannon (1948), D. Huffman (1952) a R. Fano (1961). Fanův algoritmus je z nich nejjednodušší, Shannonův algoritmus umožňuje nacházet kódování libovolně blízké optimu a konečně Huffmanův algoritmus poskytuje vždy optimální kódování. Seznámíme se s Fanovým a Huffmanovým algoritmem. Každý algoritmus slovně popíšeme a předvedeme na stejném příkladě.

Využijeme přitom grafového (stromového) znázornění úplných kódů s vlastností prefixu.

Algoritmus 1.1 *(Fanův algoritmus)*

Vstup: *Množina $X = \{x_1, \ldots, x_m\}$ s rozdělením pravděpodobností $P = (p_1, \ldots, p_m)$.*

Výstup: *Kódování K_F*

Postup: *1. Seřadíme prvky množiny X do posloupnosti $(x_{i_1}, \ldots, x_{i_m})$ podle klesající pravděpodobnosti.*

2. *Ze všech možných rozdělení této posloupnosti na dvě části (počáteční X_0 a koncovou X_1) vybereme to rozdělení, u něhož jsou součty pravděpodobností v X_0 a X_1 nejblíže, tj. hodnota*

$$\left| \sum_{x_i \in X_0} p_i - \sum_{x_i \in X_1} p_i \right|$$

je minimální. (Je-li takových rozdělení víc, vezmeme libovolné z nich.)

Prvkům z části X_0 připíšeme symbol `0` *a prvkům z části X_1 symbol* `1`.

3. *Každou vzniklou část, která obsahuje alespoň dva prvky, rozdělíme postupem z bodu 2. a opět připíšeme prvkům podle jejich umístění symbol* `0` *nebo* `1`. *Proces opakujeme tak dlouho, až se původní posloupnost rozdělí na samé jednoprvkové části.*

4. *Každému prvku $x_i \in X$ nakonec přiřadíme kód $K_F(x_i)$ tvořený posloupností symbolů, které byly postupně prvku x_i připsány.*

Příklad 1.2 *Zadáno $X = \{x_1, \ldots, x_7\}$ s rozdělením pravděpodobnosti*

$$P = (0.12, 0.20, 0.11, 0.20, 0.09, 0.09, 0.19).$$

Řešení je zachyceno na obr. 1.3 .

Střední délka sestaveného kódování K_F je 2.80.

Algoritmus 1.2 *(Huffmanův algoritmus)*

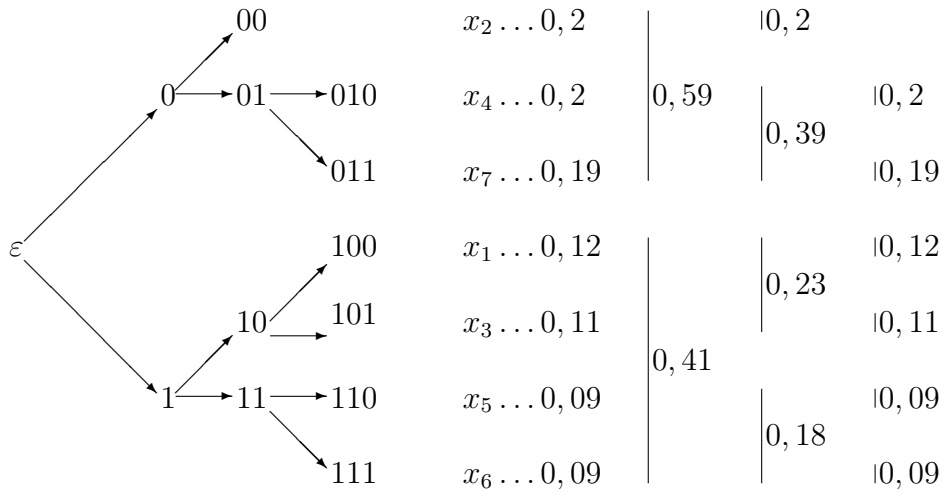
Vstup: *Množina $X = \{x_1, \ldots, x_m\}$ s rozdělením pravděpodobností $P = (p_1, \ldots, p_m)$.*

Výstup: *Kódování K_H*

Postup: 1. *Z prvků množiny X vytvoříme jednoprvkové skupiny $S_1 = \{x_1\}, \ldots, S_m = \{x_m\}$ a sestavíme soubor těchto skupin $\{S_1, \ldots, S_m\}$ s rozdělením pravděpodobností $(p_1, \ldots, p_m)$.*

2. *Následující krok provádíme opakovaně do té doby až tento soubor skupin zůstane jednočlenný:*

Najdeme dvě skupiny S', S'' se dvěma nejnižšími pravděpodobnostmi p',p''. Prvkům skupiny S' připíšeme symbol `0`, *prvkům skupiny S'' symbol* `1` *(pokud již prvky mají připsán nějaký řetězec v abecedě $\{0, 1\}$, připisujeme nové symboly dopředu). Skupiny S', S'' poté vyjmeme ze souboru skupin a nahradíme je jejich sjednocením $S = S' \cup S''$ s pravděpodobností $p = p' + p''$.*



Obrázek 1.3: Příklad k Fanovu algoritmu

3. *Každému prvku $x_i \in X$ přiřadíme slovo $K_H(x_i)$, které vzniklo postupným připisováním symbolů.*

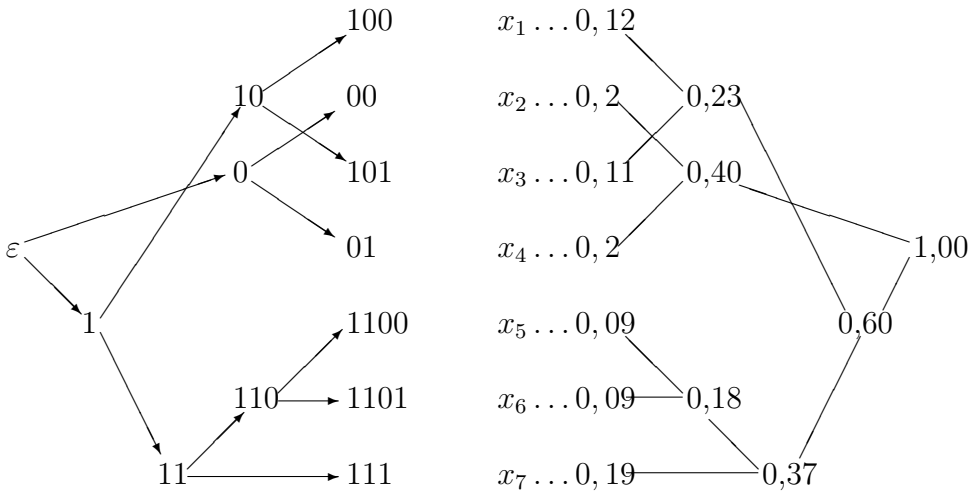
Střední délka sestaveného kódování K_H je 2.78.

Hlavní rozdíl mezi algoritmy Fana a Huffmana spočívá vlastně pouze ve směru postupu při budování kódového stromu. Fanův algoritmus buduje strom od kořene, kdežto Huffmanův algoritmus začíná od koncových vrcholů. Ukazuje se, že Huffmanův algoritmus lépe využívá specifiky daného rozdělení pravděpodobností a dospěje vždy k optimálnímu kódování.

VĚTA 1.8 *(vlastnosti Fanova a Huffmanova algoritmu)*
Pro každou množinu $X = \{x_1, \ldots, x_m\}$ a každé pravděpodobnostní rozdělení $P = (p_1, \ldots, p_m)$ se Fanův algoritmus a Huffmanův algoritmus zastaví a vydají úplná kódování s vlastností prefixu. Huffmanův algoritmus vydá vždy optimální kódování množiny X při pravděpodobnostním rozdělení P.

Fanův algoritmus vydává optimální kódování též často, avšak ne vždy. Protipříkladem je výše uvedený příklad. Přesvědčíme se o tom v tabulce 1.1 , v níž jsou zachycena pro srovnání kódování K_F,K_H a stejnoměrné kódování K_S.

Příklad 1.3



Obrázek 1.4: Příklad k Huffmanovu algoritmu

1.3 Kódy s identifikací chyb

Vysoce efektivní kódování (ve smyslu průměrné délky kódů), která jsme probírali v předchozím článku, jsou nevýhodná v případě, že při přenosu kódů dochází k chybám. Rozvinula se proto teorie takových způsobů kódování a dekódování, které umožňují zjišťovat, že došlo k chybě při přenosu kódu (kódy s detekcí chyb), nebo dokonce určit, kde a jaká chyba nastala (kódy s opravováním chyb).

Nejjednodušším příkladem kódu s detekcí chyby je paritní kontrola. Paritní (např. poslední) bit v n-bitovém řetězci je naplňován hodnotou `0` nebo `1` tak, aby celkový počet jedniček v řetězci byl vždy sudý (u sudé paritní kontroly; paritní kontrole se počet jedniček doplňuje paritním bitem vždy na lichý). Toto kódování umožňuje odhalit, že došlo (během uchovávání či přenosu kódů) v počítači či v síti k jedné záměně nuly a jedničky, tj. k chybě v jednom bitu. Neposkytuje však žádnou informaci o tom, ve kterém bitu k chybě došlo; může to dokonce být samotný kontrolní paritní bit. Dvě chyby nastalé v jednom kódu se mohou "doplňovat" tak, že výsledný řetězec bude odpovídat paritní kontrole a bude tak považován za správný kód. Poznamenejme však, že pravděpodobnost jedné bitové chyby v počítači je tak malá, že pravděpodobnost současného výskytu dvou takových chyb v jednom řetězci je zanedbatelná.

Obecně rozlišujeme chyby několika druhů:

1. *Záměna symbolů* – jeden ze symbolů je nahrazen jiným symbolem z téže

Tabulka 1.1: Srovnání středních délek sestavených kódování

	p_i	K_F	K_H	S
x_1	0,12	100	100	000
x_2	0,20	00	00	001
x_3	0,11	101	101	010
x_4	0,20	010	01	011
x_5	0,09	110	1100	100
x_6	0,09	111	1101	101
x_7	0,19	011	111	110
střední délka		2,80	2,78	3,00

Tabulka 1.2: Druhy chyb

Druh chyby	Určení chyby	Výsledný řetězec
Záměna symbolů	`0 1` na 2. místě	`0101101`
	`1 0` na 5. místě	`0001001`
Vynechání symbolu	`0` na 2. místě	`001101`
	`1` na 5. místě	`000101`
Vysunutí symbolu	`0` mezi 2. a 3. místem	`00001101`
	`1` mezi 2. a 3. místem	`00101101`
Přehození symbolů	na 3. a 4. místě	`0010101`

kódové abecedy.

2. *Vynechání symbolu* – jeden ze symbolů je vypuštěn, takže vznikne řetězec délky o jednotku kratší.

3. *Vsunutí symbolu* – před některý symbol řetězce nebo na jeho konec je přidán další symbol kódové abecedy, takže vznikne slovo o jednotku delší.

4. *Přehození symbolů* – některé dva sousední symboly si vymění místo.

Příklad 1.4 *Pro ilustraci jsou v tabulce 1.2 uvedena slova, která vzniknou ze slova* `0001101` *ve dvojkové abecedě některými chybami těchto čtyř druhů.*

Teorie kódů s detekcí a opravováním chyb je hlouběji rozpracována pro první druh chyb (záměna symbolů) ve dvojkové abecedě. Založil ji v roce

1950 R. W. Hamming. V Hammingově kódování jsou všechny kódové řetězce stejné délky n (stejnoměrné kódování), přičemž $n - k$ symbolů slova nese skutečnou informaci, zbývajících k symbolů slouží ke kontrole. Výše zmíněná paritní kontrola v počítačích je Hammingův kód pro $k = 1$. Seznámíme se se základními poznatky této teorie a na konci článku ukážeme příklad Hammingova kódu pro $n = 7$ a $k = 3$, který opravuje jednu záměnu symbolů.

Nejprve budeme přesně definovat, co se rozumí pod "detekcí a opravováním chyb".

DEFINICE 1.7 *Pro každý n-bitový řetězec w a každé přirozené s označme $Ch_s(w)$ množinu všech řetězců, které vzniknou z w po nejvýše s záměnách symbolů; pro každou množinu řetězců W pak označme $Ch_s(W)$ sjednocení všech množin $Ch_s(w)$ pro $w \in W$.*

Stejnoměrné kódování K umožňuje

1. detekci s chyb, *jestliže pro všechna $x, y \in X$, $x \neq y$ platí $K(y) \notin Ch_S(K(x))$, tj. nejvýše s záměnami symbolů v libovolném kódu prvku z X nelze obdržet jiný řetězec, který je kódem jiného prvku z X;*

2. opravení s chyb, *jestliže pro všechna $x, y \in X$, $x \neq y$ je $Ch_s(K(x)) \cap Ch_s(K(y)) = \emptyset$, tj. nejvýše s záměnami symbolů v libovolných dvou kódových řetězcích nelze obdržet stejný řetězec.*

Paritní kontrola je příkladem kódu umožňujícího detekci jedné chyby. Jedinou záměnou symbolů v řetězci, který vyhovuje paritní kontrole totiž vždy dostaneme řetězec, který jí nevyhovuje, není tedy kódovým.

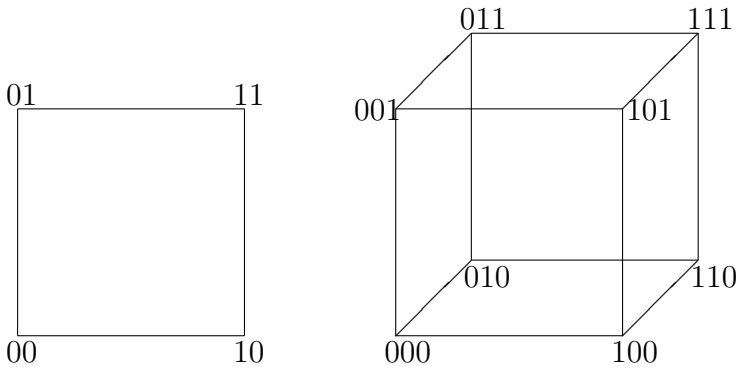
Jestliže stejnoměrné kódování K umožňuje opravení s chyb, můžeme teoreticky snadno sestrojit příslušné dekódovací zobrazení takové, že pro řetězec u je $D(u) = x$ právě tehdy, když $u \in Ch_s(K(x))$. Toto zobrazení D tedy dekóduje správně x i v případě, že při přenosu příslušného kódového řetězce $K(x)$ došlo k nejvýše s chybám (záměnám symbolů).

Nyní uvedeme základní pojem Hammingnovy teorie, který vystihuje vztah řetězců z hlediska možnosti převedení jednoho na druhý záměnami symbolů.

DEFINICE 1.8 *Jsou-li v, w řetězce délky n, pak počet míst, na kterých se v, w od sebe liší, nazýváme* Hammingova vzdálenost *v, w a značíme $h_n(v, w)$.*

Příklad 1.5 *Hammingova vzdálenost mezi* `01010` *a* `00111` *je 3, mezi* `000000` *a* `111111` *je 6. Poznamenejme, že pro každé n je $h_n(v, w) \leq n$.*

Geometricky lze Hammingovu vzdálenost znázornit pomocí n-rozměrné krychle, jejíž vrcholy jsou vhodně označeny n-bitovými řetězci. Na obrázku 1.5 jsou zachyceny případy $n = 2$ a $n = 3$. Hammingova vzdálenost mezi dvěma n-bitovými řetězci je rovna minimálnímu počtu hran, které spojují příslušné vrcholy n-rozměrné krychle. Na obrázku 1.5 se o tom pro $n = 2, 3$ snadno přesvědčíme.



Obrázek 1.5: Hammingova vzdálenost

VĚTA 1.9 *(o Hammingově metrice)*
Hammingova vzdálenost h_n je metrika na množině všech n-bitovými řetězců, tj. platí pro ni:

1. $h_n(u, v) = 0$ *právě tehdy, když* $u = w$.
2. $h_n(u, v) = h_n(v, u)$.
3. $h_n(u, v) \leq h_n(u, w) + h_n(w, v)$.

V metrickém prostoru (množině s metrikou) můžeme mluvit o okolí bodu s různým poloměrem. Pro Hammingovu metriku platí, že okolí bodu (řetězce) w o poloměru s je právě množina $Ch_s(w)$ řetězců, které z w vzniknou nejvýše s záměnami symbolů. Odtud již plyne následující věta.

VĚTA 1.10 *(podmínky pro detekci a opravování chyb)*
Pro dané stejnoměrné kódování K množiny X délky n určíme minimální Hammingovu vzdálenost

$$h_n(K) = \min\{h_n(K(x), K(y));\ x, y \in X,\ x \neq y\}.$$

Potom platí:

1. *K umožňuje detekci s chyb právě tehdy, když $s \leq h_n(K) - 1$;*

2. *K umožňuje opravení s chyb právě tehdy, když $s \leq \frac{1}{2}(h_n(K) - 1)$. .*

Příklad 1.6 *Probereme kód K_3^7 (kódování libovolné 16-ti prvkové množiny). Jedná se o Hammingův kód pro $n = 7$ a $k = 3$, tj. vlastní kódování provádí první 4 symboly, zbývající 3 symboly slouží ke kontrole.*

$x_1 \ldots$0000000
$x_2 \ldots$0001011
$x_3 \ldots$0010101
$x_4 \ldots$0011110
$x_5 \ldots$0100110
$x_6 \ldots$0101101
$x_7 \ldots$0110011
$x_8 \ldots$0111000
$x_9 \ldots$1000111
$x_{10} \ldots$1001100
$x_{11} \ldots$1010010
$x_{12} \ldots$1011001
$x_{13} \ldots$1100001
$x_{14} \ldots$1101010
$x_{15} \ldots$1110100
$x_{16} \ldots$1111111

Nejdříve vypočteme

$$h_7(K_3^7) = \min\{h_7(v, w);\ v, w \in K_3^7,\ v \neq w\} = 3,$$

Toto kódování tedy umožňuje podle věty 1.10 detekci dvou a opravování jedné chyby. Všimněme si například kódového řetězce 1100001 *a představme si, že při přenosu došlo k chybě na druhém místě – byl přijat řetězec* 1000001*. Dekódování můžeme v tomto jednoduchém případě snadno provést tak, že najdeme kódový řetězec, který je ve smyslu Hammingovy vzdálenosti přijatému nejbližší. Ukáže se, že je to* 1100001 *(vzdálenost je 1). Podobně bychom třeba* 1111011 *i* 1110111 *dekódovali jako* 1111111 *(přesněji řečeno jako prvek kódované množiny s tímto kódem).*

Sestrojit kódy pro detekci nebo opravení zadaného počtu chyb není jednoduché. Významnou metodou konstrukce těchto kódů je lineární algebra aplikovaná na dvojprvkovou množinu $\{0, 1\}$ se "sčítáním" modulo 2 (jde vlastně o bitovou operaci XOR, budeme ji značit $\oplus$) a obvyklým násobením. Mezi

n-bitovými řetězci se tyto operace provádějí "po bitech". Lineární podprostory tohoto lineárního prostoru n-bitových řetězců se pak nazývají lineární kódy. Např. Hammingův kód z Příkladu 1.6 je lineární ("součet" každých dvou kódových řetězců je opět kódový řetězec). S lineárními kódy lze takto pracovat běžnými algebraickými prostředky. Kódování a dekódování (detekce a oprava chyb) se realizuje pomocí speciálních matic.

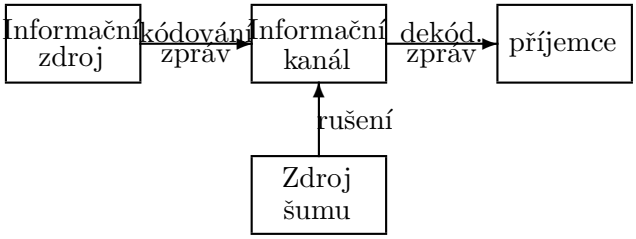
1.4 Přenos informací

V tomto článku se budeme zabývat hlavní oblastí aplikace teorie kódování – komunikačními systémy pro přenos informací. Matematickou teorii komunikace (zvanou též teorie informace) předložil v roce 1948 C. Shannon.

Důležité je upřesnit, v jakém smyslu budeme používat pojmy komunikace a informace. Rozlišují se tři úrovně komunikačních problémů:

- technická, týkající se přesného a rychlého přenosu formálně upravené (symbolicky zapsané) zprávy;
- sémantická, týkající se přenosu zamýšleného významu zprávy;
- pragmatická, týkající se užitečnosti zprávy pro příjemce nebo dopadu jejího přijetí na jeho chování.

Matematická teorie komunikace se zabývá technickou úrovní komunikace. Zprávy jsou studovány z hlediska pravděpodobnosti jejich výskytu, ne z hlediska jejich smyslu, významu, věrohodnosti a efektu.



Obrázek 1.6: Schéma komunikačního systému

Obecné schéma komunikačního systému je naznačeno na obr. 1.6 .

Základním problémem teorie komunikace je návrh postupů kódování a dekódování tak, aby byl maximalizován rozsah informací přenositelných komunikačním systémem, speciálně za přítomnosti šumu. Shannonovy výsledky

ukazují závislost množství a rychlosti přenosu informací na vydatnosti informačního zdroje a kapacitě informačního kanálu. Abychom mohli formulovat (alespoň zjednodušeně) základní věty teorie komunikace, musíme nejdříve definovat pojmy míra informace obsažená ve zprávě, entropie informačního zdroje a kapacity informačního kanálu.

Numerická míra množství informace obsažené v konkrétní zprávě musí být funkcí pravděpodobnosti výskytu této zprávy (neboť z hlediska teorie komunikace není kromě pravděpodobnosti výskytu o obsahu zprávy nic jiného známo). Intuitivně je přirozená představa, že přijetí méně pravděpodobné zprávy přinese více informace. Oprávněné je též očekávat, že množství informace obsažené ve dvou na sobě nezávislých zprávách je součtem množství informací obsažených v každé z nich. Jestliže budeme navíc ještě požadovat, aby míra informace v jisté zprávě (zprávě vyskytující se s pravděpodobností 1) byla nulová a závislost míry informace na pravděpodobnosti výskytu zprávy byla spojitá, zůstanou jako vyhovující pro míru informace pouze logaritmické funkce. Ukazuje to následující věta.

VĚTA 1.11 *(o Hartley-Shannonově formuli) Nechť f je reálná funkce definovaná na intervalu $(0, 1]$ s těmito vlastnostmi:*

1. *f je klesající,*
2. *$f(p_1 \cdot p_2) = f(p_1) + f(p_2)$,*
3. *$f(1) = 0$,*
4. *f je spojitá.*

Potom $f(p) = -c \cdot \log_a p$, kde $a > 1$, $c > 0$.

V konkrétní definici míry informace byly nakonec přijaty parametry $a = 2$, $c = 1$. Odpovídá to požadavku, aby míra informace obsažené ve zprávě, která má pravděpodobnost výskytu $\frac{1}{2}$ byla jednotková. Pro tuto jednotku informace se pak používá označení 1 bit (binary digit). Míru informace budeme definovat přímo v souvislosti s informačním zdrojem.

DEFINICE 1.9 Informační zdroj (X, P) *je konečná množina zpráv $X = \{x_1, \ldots, x_m\}$ s rozdělením pravděpodobnosti $P = (p_1, \ldots, p_m)$.*

Míra informace $I(x)$ *zprávy $x \in X$ s pravděpodobností p je definována jako $I(x) = -\log_2 p$.*

Střední míra informace *(entropie) $H(X, P)$ informačního zdroje (X, P) je vážený průměr měr informace všech zpráv zdroje:*

$$H(X, P) = -\sum_{i=1}^{m} p_i \log_2 p_i.$$

Tabulka 1.3: Příklad pro výpočet míry informace a entropie

(X, P)	p_i	$-\log_2 p_i$	$-p_i \cdot \log_2 p_i$
x_1	0,12	3,0592	0,3671
x_2	0,20	2,3222	0,4644
x_3	0,11	3,1847	0,3503
x_4	0,20	2,3222	0,4644
x_5	0,09	3,4743	0,3127
x_6	0,09	3,4743	0,3127
x_7	0,19	2,3962	0,4533
		$H(X, P) =$	2,7269

Příklad 1.7 *Na ukázku vypočteme míru informace a entropie pro rozdělení pravděpodobnosti, které známe z příkladů 1.2 a 1.3. Zadání i řešení příkladu zapíšeme v tabulce 1.3 .*

Uvedená definice informačního zdroje odpovídá nejjednoduššímu případu, kdy jsou zdrojem produkovány v diskrétním čase zprávy z konečné množiny, přičemž pravděpodobnost výskytu každé zprávy se nemění ani s časem, ani v závislosti na výskytech jiných zpráv. Obecnější definice však vedou ke značnému zkomplikování všech úvah.

Definice střední míry informace pochází od Shannona. Vychází z chápání míry informace jako náhodné veličiny na množině zpráv X. Střední míra informace je pak střední hodnota této diskrétní náhodné veličiny. Střední míru informace informačního zdroje lze zavést také axiomaticky, podobně jako u míry informace. Samotnou míru informace zprávy v takovém případě není třeba zavádět.

Termín entropie, který se pro střední míru informace často používá, je převzat z klasické termodynamiky, kde entropie popisuje stupeň neuspořádanosti v systému. Komunikační entropie v tomto smyslu měří nejistotu spojenou s informačním zdrojem před tím, než je přijata zpráva. Tato nejistota je největší tehdy, když všechny zprávy jsou stejně očekávané, mají stejnou pravděpodobnost.

Pro takové rovnoměrné rozdělení pravděpodobnosti výskytu m zpráv, kde každá má pravděpodobnost $\frac{1}{m}$, je entropie rovna:

$$-\sum_{i=1}^{m} \frac{1}{m} \log_2 \frac{1}{m} = -\log_2 \frac{1}{m} = \log_2 m.$$

VĚTA 1.12 *(o maximální entropii)*
Je-li (X, P) *informační zdroj o* m *různých zprávách, pak*

$$0 \leq H(X, P) \leq \log_2 m,$$

tj. $\log_2 m$ *je maximální entropie informačního zdroje o* m *zprávách.*

Maximální entropie informačního zdroje "typu ano/ne" se dvěma možnými zprávami je tedy rovna 1 bitu. Maxima se nabývá v případě, kdy obě možnosti mají stejnou pravděpodobnost $\frac{1}{2}$.

Často se zavádí další pojmy odvozené z pojmu entropie. *Relativní entropie* $H_r(X, P)$ je definována jako podíl skutečné entropie (střední míry informace) k maximální možné, tj. pro informační zdroj s m zprávami

$$H_r(X, P) = \frac{H(X, P)}{log_2 m}.$$

Potom podle věty 1.12 je $0 \leq H_r(X, P) \leq 1$. Rozdíl $1 - H_r(X, P)$ bývá nazýván *redundance* informačního zdroje.

Příklad 1.8 *Při hodnotách z výše uvedeného zadání (tab. 1.3 dostáváme hodnotu redundance informačního zdroje rovnu* $1 - H_r(X, P) = 1 - 0.9713 = 0.0287$.

Informační zdroj (X, P) produkuje v čase posloupnost zpráv z X. Pro rozdělení pravděpodobnosti produkovaných posloupností platí obdoba zákona velkých čísel: K libovolně malému $\varepsilon > 0$ existuje takové přirozené číslo $n(\varepsilon)$, že pro všechna $n \geq n(\varepsilon)$ je možno množinu všech posloupností zpráv délky n rozdělit na dvě části:

- na množinu asi $2^{n \cdot H(X,P)}$ "důležitých" posloupností, které mají všechny přibližně tutéž pravděpodobnost výskytu a dohromady mají pravděpodobnost výskytu alespoň $1 - \varepsilon$;
- na zanedbatelný zbytek "nedůležitých" posloupností, které mají dohromady pravděpodobnost menší než ε.

Přesnou, ale neprůhlednou formulaci tohoto tvrzení přináší následující věta.

VĚTA 1.13 *(Shannon-McMillanova věta o rozdělení produkovaných slov)*
Nechť (X, P) *je informační zdroj. Potom k libovolnému reálnému* $\varepsilon > 0$ *existuje* $n(\varepsilon)$ *takové, že pro všechna přirozená* $n \geq n(\varepsilon)$ *je*

$$P\left(\left\{y_1 \dots y_n \in X^n;\ \left|\frac{1}{n}\sum_{j=1}^{n} \log_2 P(y_j) + H(X, P)\right| \geq \varepsilon\right\}\right) < \varepsilon.$$

Informační kanál je fyzikální médium pro přenos signálů. Kanál přijímá vstupní signály a vydává výstupní signály. Budeme předpokládat, že na vstupu kanál přijímá a na výstupu vydává právě n signálů v časové jednotce odpovídající jedné zprávě. V ideální situaci jsou vstupní a výstupní signál vždy totožné, ale ve skutečnosti je takřka vždy přítomen šum, který může způsobit na výstupu chybu.

Přítomnost šumu při přenosu informačním kanálem je v jednodušším případě charakterizována maticí pravděpodobností přechodu mezi signály na vstupu a výstupu. Pro každý signál je v ní uvedeno rozdělení pravděpodobnosti výskytu různých signálů na výstupu, jestliže tento signál byl na vstupu.

Podobně jako jsme číselně charakterizovali vydatnost informačního zdroje, lze charakterizovat přenosovou schopnost informačního kanálu. Tuto kapacitu informačního kanálu je možno definovat různými způsoby:

- pomocí maximálního množství informace, které je schopen přenést za časovou jednotku;
- pomocí maximální střední míry informace informačních zdrojů, jejichž zprávy je kanál schopen spolehlivě přenášet;
- pomocí počtu spolehlivě přenositelných posloupností.

Pro informační kanály, s nimiž se v praxi setkáváme, vedou tyto tři přístupy ke stejné hodnotě kapacity kanálu.

Informační kanály a jejich kapacitu nebudeme přesně definovat ve vší obecnosti, ale soustředíme se pro jednoduchost na nejjednodušší případ informačního kanálu, který přenáší signály 0,1; každý z nich je schopen přenést ve stejném čase a případné rušení se projevuje stejně pro oba signály (symetrie).

DEFINICE 1.10 Binární informační kanál *s pravděpodobností chyby p přenáší signály* 0,1 *se šumem, který je charakterizován touto maticí pravděpodobnosti přechodu:*

	0	1
0	$1-p$	p
1	p	$1-p$

Kapacita *binárního kanálu s pravděpodobností chyby p je pak určena takto:*

$$C(p) = 1 + p \cdot \log_2 p + (1-p) \cdot log_2(1-p).$$

Tato definice zahrnuje i případ binárního informačního kanálu bez šumu – pro $p = 0$ dostaneme kapacitu $C(0) = 1$. Druhý extrémní případ nastane, když $p = \frac{1}{2}$ (správný přenos je stejně pravděpodobný jako chybný, výstup je zcela nezávislý na vstupu). Dosazením do vzorce zjistíme, že $C(\frac{1}{2}) = 0$, tj. kanál pak nemá žádnou kapacitu. V ostatních případech leží kapacita binárního kanálu mezi 0 a 1, např.

$$C(0.1) = 1 + 0.1 \log_2 0.1 + 0.9 \log_2 0.9 = 0.531.$$

Nyní již můžeme přikročit k formálnímu popisu jisté třídy jednoduchých komunikačních systémů.

Ve zjednodušeném pojetí komunikačního systému budeme předpokládat, že informační zdroj vysílá zprávy v pravidelných časových intervalech a nezávisle na předchozích vyslaných zprávách.

Definici kódování zpráv zobecníme pro případ komunikačních systémů takto: Místo kódování jedné zprávy budeme uvažovat o kódování celých sekvencí zpráv. Prosté zobrazení, které každé posloupnosti k zpráv přiřazuje řetězec symbolů `0/1` délky d, budeme nazývat (k, d)-*kódování*.

DEFINICE 1.11 Jednoduchý binární komunikační systém *je čtveřice:*

- *informační zdroj* (X, P)*, který v pravidelných časových intervalech produkuje nezávislé zprávy z množiny zpráv* $X = \{x_1, \ldots, x_m\}$ *s rozdělením pravděpodobnosti* $P = (p_1, \ldots, p_m)$*;*

- (k, d) *– kódování, které* k*-ticím po sobě jdoucích zpráv přiřazuje posloupnost signálů* `0/1` *délky* d*;*

- *binární informační kanál s pravděpodobností chyby* p*, který za dobu vyslání jedné zprávy přenese* n *signálů* `0/1`*;*

- *dekódovací zobrazení.*

Pro daný komunikační systém lze na základě pravděpodobnosti chyby informačního kanálu, rozdělení pravděpodobnosti zpráv a vlastností použitého kódování určit *pravděpodobnost chybného přenosu zpráv* tímto komunikačním systémem, tj. pravděpodobnost, že zpráva přijatá po dekódování výstupu z informačního kanálu se bude lišit od zprávy, která byla na vstup kanálu původně zakódována.

O tom, zda lze pravděpodobnost chyby při přenosu minimalizovat vhodným kódováním, se teoreticky vyjadřuje následující věta:

VĚTA 1.14 *(Shannonova přímá a obrácená věta o možnosti přenosu) Nechť je dán informační zdroj* (X, P) *se střední mírou informace* $H = H(X, P)$ *a binární informační kanál s kapacitou* $C = C(p)$*, který za dobu vyslání jedné zprávy přenese* n *signálů* `0`/`1`*. Potom platí:*

1. *Jestliže* $n \geq \frac{H}{C}$*, pak ke každé zadané libovolně malé chybě přenosu existuje* (k, d) *– kódování a dekódování tak, že vzniklý komunikační systém má pravděpodobnost chybného přenosu zpráv menší než je zadaná.*

2. *Jestliže* $n < \frac{H}{C}$*, pak existuje kladná mez taková, že pro libovolné* (k, d)*-kódování a dekódování je ve vzniklém komunikačním systému pravděpodobnost chybného přenosu zpráv rovna přinejmenším této mezi.*

Libovolnou přesnost přenosu lze tedy zajistit jedině v případě, že informační kanál je schopen ke každé zprávě přenést (v dané časové jednotce odpovídající jedné zprávě) alespoň $\frac{H}{C}$ nulajedničkových signálů.

Příklad 1.9 *Pro informační zdroj, který má střední míru informace* $H = 2,7269$ *a binární informační kanál s pravděpodobností chyby* 0.1*, který má kapacitu* $C = 0,5310$*, dostaneme*

$$\frac{H}{C} = 5.1354,$$

takže hranicí pro délku n *je 6. Pro* $n \geq 6$ *tedy existuje možnost přenášet zprávy daného zdroje daným kanálem s libovolně malou pravděpodobností chybného přenosu pomocí vhodně zkonstruovaného* (k, d)*-kódování a dekódování. Pro* $n < 6$ *tato možnost neexistuje.*

1.5 Cvičení ke kapitole 1

1. Posuďte vlastnosti (prefixu, separability, úplnosti) u následujících kódování:

a)		b)		c)	
	$x_1 \ldots$ `0`		$y_1 \ldots$ `0`		$z_1 \ldots$ `10`
	$x_2 \ldots$ `010`		$y_2 \ldots$ `10`		$z_2 \ldots$ `101`
	$x_3 \ldots$ `101`		$y_3 \ldots$ `110`		$z_3 \ldots$ `001`
			$y_4 \ldots$ `111`		

2. Sestrojte efektivní kódování podle Fanova algoritmu a Huffmanova algoritmu pro následující informační zdroje

$$P_1 = (0.25, 0.25, 0.20, 0.15, 0.10, 0.05)$$

$$P_2 = (0.02, 0.10, 0.01, 0.01, 0.20, 0.30, 0.15, 0.14, 0.05, 0.02)$$

3. Určete schopnost detekce a opravování chyb u kódování:

$$
\begin{aligned}
x_1 &\ldots \texttt{000000}\\
x_2 &\ldots \texttt{111000}\\
x_3 &\ldots \texttt{110011}\\
x_4 &\ldots \texttt{001011}\\
x_5 &\ldots \texttt{101101}\\
x_6 &\ldots \texttt{010101}\\
x_7 &\ldots \texttt{011110}\\
x_8 &\ldots \texttt{100110}
\end{aligned}
$$

   Dekódujte přijaté řetězce `000001`, `000111`, `100001`.

4. Vypočtěte střední míru informace (entropii), relativní entropii a redundanci u informačních zdrojů z cvičení 2.

5. Aplikujte Shannonovy věty na informační zdroje z cvičení 2 a tyto binární informační kanály:

   - bez šumu (zde srovnejte s výsledky cvičení 2)
   - s pravděpodobností chyby p=0.2.

Kapitola 2

Základy kryptografie

V této kapitole popíšeme vybrané metody šifrování, což jsou metody kódování sloužící k utajení obsahu zprávy během jejího přenosu. Hlavní pozornost budeme věnovat moderním metodám šifrování při digitálním přenosu dat; k uvedení do problematiky však neopomineme ani některé klasické metody šifrování používané v minulých stoletích a tisíciletích.

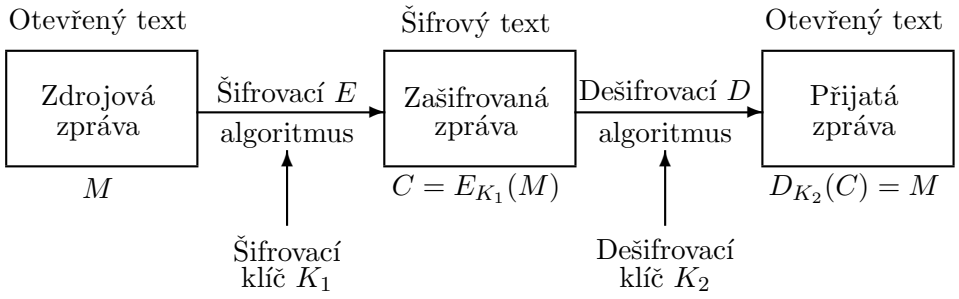
Podrobnosti o metodách šifrování lze nalézt v řadě učebnic a monografií, např. [1, 12, 8, 16, 15, 18, 3, 5, 19], z nichž také ve výkladu vycházíme.

Systematicky se utajováním zpráv zabývá *kryptografie*, která je nerozlučně doprovázena *kryptoanalýzou* – oborem zaměřeným na odhalování způsobu šifrování, prolamování šifer a odkrývání obsahu utajených zpráv. Výzkum v oblasti kryptografie a kryptoanalýzy je náplní *kryptologie*.

Zpráva, která má být utajována, je nejčastěji řetězec znaků nějaké abecedy; budeme ji nazývat *otevřený text*. Tímto textem mohou být také digitalizované obrázky, zvukové signály či videoklipy. *Kryptosystém* (viz obr. 2.1), který slouží k utajování zpráv, je tvořen *šifrovacím algoritmem* E (z anglického *encryption* nebo též *enciphering*), jehož parametrem je *šifrovací klíč* K_1, a *dešifrovacím algoritmem* D, jehož parametrem je *dešifrovací klíč* K_2.

Otevřený text M je v kryptosystému převeden na *šifrový text* $C = E_{K_1}(M)$, který je přenášen komunikačním kanálem k příjemci. V této kapitole budeme pro jednoduchost předpokládat, že přenos je bezchybný (v případě kanálů s pravděpodobností výskytu chyb je šifrový text ještě kódován vhodným samoopravným kódem). Příjemce obdrží šifrový text C a provede dešifrování $D_{K_2}(C)$. Pro každou utajenou zprávu M musí pochopitelně v korektním kryptosystému platit $D_{K_2}(E_{K_1}(M)) = M$.

V tomto schématu předpokládáme, že přenosový kanál nelze zcela ubránit před *odposlechem*, tj. šifrový text C může být k dispozici nejen určenému příjemci, nýbrž i *útočníkovi*, který se jej snaží kryptoanalytickými meto-



Obrázek 2.1: Schéma kryptosystému.

dami dešifrovat. Historie ukázala, že je nutno předpokládat i to, že útočník zná šifrovací a dešifrovací algoritmus. Bezpečnost kryptosystému je tak dána především obtížností nalezení dešifrovacího klíče K_2.

V konvenčních *symetrických kryptosystémech* je dešifrovací klíč K_2 stejný jako šifrovací klíč K_1, nebo lze K_2 ze znalosti K_1 snadno určit. Šifrovací klíč K_1 je proto nutno utajit – mluvíme o *šifrování s tajným klíčem*. Podstatou *asymetrických kryptosystémů*, vyvíjených od 70. let minulého století, je použití různých šifrovacích a dešifrovacích klíčů, a to takových, že dešifrovací klíč K_2 nelze snadno určit ze šifrovacího klíče K_1 (složitost potřebného výpočtu je dostupnými prostředky nezvládnutelná). Pak není nutno šifrovací klíč tajit, naopak je možné jej zveřejnit pro všechny, kteří chtějí posílat příjemci šifrové zprávy – mluvíme o *šifrování s veřejným klíčem*.

Kryptoanalytické metody prolamování šifer nevycházejí jen ze znalosti kryptosystému a zachyceného šifrového textu, ale snaží se získat a využít různé další poznatky. Podle jejich rozsahu se rozlišují typy *kryptoanalytických útoků* založených na specifických znalostech dostupných útočníkovi. Možnosti útočníka se stupňují, jestliže má k dispozici: více šifrových textů, k nim i zdrojových otevřených textů, výsledky zašifrování zadaných otevřených textů, informace z průběhu šifrování adaptivně zadávaných otevřených textů, výsledky dešifrování zadaných šifrových textů.

Obzvláště výhodná situace pro útočníka nastává, dojde-li k částečnému či úplnému prozrazení dešifrovacích klíčů (postupům jak dosáhnout prozrazení klíčů se ovšem v tomto textu věnovat nebudeme).

Dříve, než se pustíme do popisu vybraných moderních kryptosystémů, začneme s trochou historie. To především proto, abychom si přiblížili základní myšlenky kryptoanalýzy.

2.1 Vigenèrova substituční šifra

Jednoduchým způsobem šifrování, které používali již staří Římané, je nahrazovaní jednotlivých znaků otevřeného textu jinými znaky (stejné nebo jiné šifrové abecedy). Jedná se vlastně o prosté kódování vstupní abecedy[1]. Pokud je na celý otevřený text používána pouze jedna substituce, mluvíme o *monoalfabetickém šifrování*; pokud se substituce mění např. podle pozice znaku v textu, mluvíme o *polyalfabetickém šifrování*.

Příklad 2.1 Monoalfabetická Césarova substituční šifra. *Tímto názvem je obvykle označována šifra, při jejímž použití jsou písmena uspořádané abecedy o m znacích nahrazována písmeny následujícími v abecedě o daný počet míst dále. Uvažujeme samozřejmě cyklický posun. Klíčem je tedy délka cyklického posunu* k, $1 \leq k < m$.

Při použití anglické abecedy dostáváme např. při posunu $k = 7$ *substituci*

```
abcdefghijklmnopqrstuvwxyz
HIJKLMNOPQRSTUVWXYZABCDEFG
```

Je-li šifrový text výsledkem aplikace monoalfabetické substituce na otevřený text, který je napsán v jazyce se známými frekvencemi výskytů jednotlivých písmen v běžných textech, můžeme kryptoanalýzu založit na porovnání frekvencí výskytů (*frekvenční kryptoanalýza*). V šifrovém textu najdeme znaky s vysokou relativní frekvencí výskytů a předpokládáme, že jsou to šifry odpovídající písmenům vstupní abecedy s podobnou relativní frekvencí výskytu v textech. Postupně se pokoušíme stanovit přiřazení dalších písmen a přitom kontrolujeme, zda dešifrováním vzniká smysluplný otevřený text.

Příklad 2.2 *Ilustrujme si kryptoanalýzu Césarovy šifry na krátkém šifrovém textu, o kterém víme, že otevřený text je v angličtině (relativní frekvence výskytu 26 písmen bez mezery v anglických textech obsahuje tabulka 2.1 – podle [2]):*

```
LFDPHLVDZLFRQTXHUHG
```

Při řešení této úlohy využíváme znalost toho, že maximální frekvenci v šifrovém textu mají H a L, které by tak mohly být šiframi nejčastěji se vyskytujících písmen e, t, a, o, ... Zkusíme první nabízející se možnost, kdy H je substitucí písmene e, tj. posun $k = 3$. *Již na první pokus získáváme*

[1] V ilustračních příkladech budeme v otevřeném textu používat malá písmena bez mezer a jako šifrovou abecedu velká písmena.

Tabulka 2.1: Relativní frekvence písmen v anglických textech

Znak	*frekvence (%)*	*Znak*	*frekvence (%)*
a	8,167	n	6,749
b	1,492	o	7,507
c	2,782	p	1,929
d	4,253	q	0,095
e	12,702	r	5,987
f	2,228	s	6,327
g	2,015	t	9,056
h	6,094	u	2,758
i	6,966	v	0,978
j	0,153	w	2,360
k	0,772	x	0,150
l	4,025	y	1,974
m	2,406	z	0,074

dešifrovaný text: `icameisawiconquered` *je slavný Césarův výrok, napsaný v angličtině bez mezer.*

Poznamenejme, že kryptoanalýzu Césarovy šifry můžeme provést snadno i "hrubou silou" – prověřením všech 25 netriviálních klíčů. V případě obecné monoalfabetické substituční šifry by však šlo o celkem $26!-1$ *různých možných permutací anglické abecedy.*

Určitou obranu proti frekvenční kryptoanalýze poskytuje *homofonní substituční šifra*, kdy vysoce frekventovaným písmenům přiřazujeme více možných šifer (např. dvojciferných čísel), kdežto málo frekventovaným pouze jednu šifru (jedno dvojciferné číslo).

Polyalfabetické substituční šifry jsou vesměs založeny na spojení více monoalfabetických šifer. Základním typem je *Vigenèrova šifra*, což je v minulých stoletích rozšířený polyalfabetický substituční kryptosystém tvořený postupným periodickým používáním souboru d Césarových šifer (s různými posuny $k_1, \ldots, k_d$) na jednotlivé znaky otevřeného textu. Vigenèrova šifra tedy umožňuje řádově zvětšovat počet možných klíčů prodlužováním *periody* d.

Klíč: $K = (k_1, \ldots, k_d)$ je zadáván jako *heslo* (slovo délky d, jehož jednotlivá písmena určují posuny).

Šifrování: Pro $i = 1, \ldots, d$ jsou písmena otevřeného textu na pozicích i,

$i+d, i+2d, i+3d, \ldots, i+l \cdot d$ šifrována pomocí Césarovy šifry s posunem k_i.

Dešifrování: Opačné posuny.

V praxi byl používán tzv. *Vigenèrův čtverec* (tabulka 2.2), který umožňoval rychlé ruční šifrování a dešifrování.

Tabulka 2.2: Vigenèrův čtverec

	Znak otevřeného textu																								
Klíč	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Příklad 2.3 *Ukázka Vigenèrovy šifry s heslem* host *(perioda 4,* $k_1 = 7, k_2 = 14, k_3 = 17, k_4 = 18$*):*

otevřený text:	`individualcharacter`
heslo:	`hosthosthosthosthos`
šifrový text:	`PBVBCWVNHZVAHFSVASJ`

Kryptoanalýza Vigenèrovy šifry se skládá ze dvou částí:

a) Hledání periody d, které je založeno na *analýze koincidencí v šifrovém textu* (W. Friedman, 1920) a/nebo *analýze odstupu stejných posloupností* v šifrovém textu (Babbage 1854, Kasiski 1863).

b) Provedení d frekvenčních kryptoanalýz (pro $i = 1, \ldots, d$): Zjistíme frekvence výskytu šifer na pozicích $i, i+d, i+2d, \ldots, i+l \cdot d$ šifrového textu a odhadneme příslušný posun k_i.

Podrobněji se budeme těmito postupy zabývat v následujících odstavcích.

2.1.1 Kryptoanalýza s využitím koincidencí

Analýza koincidencí slouží k určení periody Vigenèrovy šifry rozborem dostatečně dlouhého šifrového textu.

Uvažujeme-li dvě zcela náhodné posloupnosti písmen mezinárodní latinské abecedy (tj. nezávisle generované s rovnoměrným rozložením pravděpodobností výskytu 26 písmen nezávisle na pozici), pak pravděpodobnost, že se na jedné pozici objeví v obou posloupnostech stejné písmeno, je

$$26 \cdot \left(\frac{1}{26}\right)^2 = \frac{1}{26} = 0,038\ldots,$$

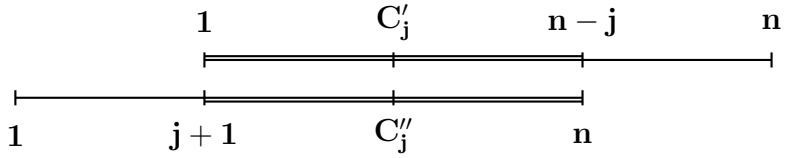
neboť pravděpodobnost výskytu konkrétního písmene je $\frac{1}{26}$, pravděpodobnost výskytu konkrétního identického páru písmen je $\left(\frac{1}{26}\right)^2$ a počet možných identických párů písmen je 26.

Jsou-li však obě posloupnosti písmen např. anglickými texty, pak pravděpodobnost, že se na stejné pozici objeví v obou posloupnostech stejné písmeno, je přibližně

$$\sum_{l=1}^{26} (\hat{p}(x_\ell))^2 = 0,065\ldots,$$

kde $\hat{p}(x_\ell)$ je relativní frekvence výskytu písmene x_ℓ v anglických textech (viz Tabulka 2.1), a $(\hat{p}(x_\ell))^2$ je tedy odhad pravděpodobnosti výskytu písmene x_ℓ na stejných pozicích ve dvou textech.

Poznamenejme na tomto místě, že uvedená hodnota, označovaná jako *míra koincidence*, se pro jednotlivé jazyky liší, např. pro němčinu je 0.077, pro češtinu 0.058, pro malajštinu 0.085. Používá se proto i k odhadování, v jakém jazyce je napsán daný otevřený text.



Obrázek 2.2: Výpočet koincidence v šifrovém textu.

Mějme nyní šifrový text C délky n. Posuneme jeho dvě kopie vůči sobě o j znaků (viz obr. 2.2) a označíme:

$C'_j = C[1, n-j]$ – počáteční část šifrového textu do pozice $n-j$,

$C''_j = C[j+1, n]$ – koncovou část šifrového textu od pozice $j+1$.

Pak obě části C'_j, C''_j mající délku $n-j$ představují zašifrování otevřeného textu v daném jazyce a platí:

1. Je-li j rovno periodě šifry (nebo jejímu násobku), pak na každé pozici jsou písmena v C'_j, C''_j stejně zašifrovaná, takže koincidence jejich šifer odpovídá koincidenci písmen v otevřeném textu (resp. koincidenci mezi jeho částmi M'_j a M''_j).

2. Jestliže j není perioda (ani její násobek), pak jsou na každé pozici použity jiné substituce, a tedy koincidence písmen v C'_j, C''_j jsou spíše náhodné.

Z výše uvedeného vyplývá následující postup aplikace analýzy koincidencí při hledání periody d:

- pro $j = 1, 2, 3, \ldots, n_0$ (kde $n_0 < \frac{1}{2}n$) zjistíme relativní počet koincidencí RC_j mezi výše definovanými částmi šifrového textu C'_j, C''_j.

- analyzujeme posloupnost čísel $RC_1, RC_2, RC_3, \ldots, RC_{n_0}$:
  vyšší čísla RC_j (blízká míře koincidence daného jazyka) naznačují, že j je periodou, nižší čísla (blízká 0.038) naznačují opak.

- vezmeme minimální d, pro něž RC_d naznačují periodu, a přesvědčíme se, že často též čísla $RC_{r \cdot d}$ pro $r = 2, 3, \ldots$ naznačují periodu a ostatní naopak často ne.

Příklad 2.4 *Kryptoanalýzu Vigenèrovy šifry budeme ilustrovat na příkladu, v němž je zadán šifrový text, vzniklý Vigenèrovou šifrou z českého textu s vynecháním diakritiky a mezer:*

```
CYLSOMDSORRICEFUKMOXCVXSVOOZVJXMOJXMTEWDXZGDTLXOCIVGKLXLEZXL
ORBBDNXCYSMHJNRBRVLDMHGZYDWDVEGHZRHOBOMHCPBNXASMKSBBRNXOBAMD
VSDXMHLSKBNLYHHTTIMHXAAQXEVJNYUXCEKNJKKZTEEHXEIQOCMNENTROSBE
BYCDCTMNXEVNJCXKKNHUOHHSITHRSFKXXEFZTIIQODVGEDVD
```

V tab. 2.3 jsou spočteny koincidence pro $j = 1, \ldots, 36$ *(*AC_j *značí absolutní počet koincidencí,* RC_j *relativní). Míra koincidence v českých textech je přitom* 0.058*.*

Všimneme si, že relativní počet koincidencí RCj je větší než $0,05$ *pro* $j = 1, 4, 6, 8, 10, 14, 17, 18, 20, 24, 26, 32, 35, 36$*; // naopak menší než* $0,04$ *pro* $j = 2, 3, 5, 7, 13, 15, 16, 19, 21, 22, 23, 27, 28, 29, 31, 33, 34$*. Nabízí se závěr, že perioda je* $4, 6$*, nebo* 8*.*

Tabulka 2.3: Koincidence k příkladu 2.4

j	AC_j	RC_j	j	AC_j	RC_j	j	AC_j	RC_j
1	13	0,0572	13	8	0,0372	25	7	0,0344
2	8	0,0353	14	15	0,0700	26	12	0,0594
3	8	0,0355	15	8	0,0375	27	8	0,0398
4	17	0,0758	16	8	0,0377	28	8	0,0399
5	7	0,0313	17	11	0,0521	29	5	0,0251
6	13	0,0585	18	14	0,0666	30	8	0,0404
7	7	0,0316	19	8	0,0382	31	6	0,0304
8	17	0,0772	20	15	0,0721	32	12	0,0612
9	9	0,0410	21	8	0,0386	33	6	0,0307
10	11	0,0504	22	3	0,0145	34	7	0,0360
11	9	0,0414	23	5	0,0243	35	11	0,0569
12	10	0,0462	24	17	0,0833	36	13	0,0677

2.1.2 Kasiského kryptoanalýza

Kasiského přístup ke kryptoanalýze[2] Vigenèrovy šifry vychází z úvahy, že dvě stejné posloupnosti písmen v šifrovém textu zřejmě vznikly, až na výjimky, ze stejného slova (posloupností písmen) vyskytujícího se dvakrát v otevřeném textu a zašifrovaného v obou případech stejně. Odtud plyne, že odstup dvou stejných posloupností v šifrovém textu je většinou násobkem periody d.

V šifrovém textu proto zjistíme odstupy stejných posloupností písmen (alespoň 3-členných) a určíme jejich společného dělitele (zřejmé výjimky zanedbáme).

Příklad 2.5 *Metodu si budeme ilustrovat na šifrovém textu z příkladu 2.4. Zde se dvakrát vyskytují tyto tři trojpísmenné posloupnosti:*

Posloupnost	*Odstup*
`JXM`	4
`XEV`	44
`IQO`	52

Odtud odhadujeme délku periody $d = 4$, neboť společným dělitelem odstupů je 4. Tato perioda je též mezi odhadovanými na základě analýzy koincidencí v příkladu 2.4.

[2]Publikován Kasiským v roce 1863; již v roce 1854 jej vyvinul Babbage, který jej však nepublikoval z důvodů používání Vigenèrovy šifry britskou armádou v probíhající Krymské válce.

2.1.3 Frekvenční kryptoanalýza při znalosti periody

Známe-li periodu d Vigenèrovy šifry, redukuje se její další kryptoanalýza na frekvenční analýzu d Césarových šifer používaných postupně s periodou d. Vybereme proto pro každé $i = 1, \ldots, d$ ze šifrového textu podposloupnost písmen na pozicích $i, i + d, i + 2d, \ldots, i + l \cdot d$ a provedeme pro ni rozbor frekvencí výskytu znaků. Na základě srovnání s tabulkou frekvencí písmen v příslušném jazyce odhadujeme možné posuny k_i použité Césarovy šifry. S takto odhadnutými klíči $(k_1, \ldots, k_d)$ provádíme pokusné dešifrování.

Příklad 2.6 *Šifrový text z příkladu 2.4, u něhož jsme předchozími analýzami odhadli periodu $d = 4$, rozdělíme do 4 podposloupností obsahujících znaky, které byly kódovány s použitím stejného posunu, i-tá podposloupnost tedy bude obsahovat znaky z šifrového textu v pozicích $i, i+4, i+8, \ldots, i+4 \cdot l$ šifrového textu. Tyto posloupnosti jsou zaznamenány v tabulce 2.4.*

Nyní srovnáme frekvence u každé posloupnosti s tab. 2.5 frekvencí písmen v češtině (viz [12]). Můžeme to provést víceméně zkusmo tím, že najdeme nejfrekventovanější písmeno šifrového textu a přiřadíme jej k nejfrekventovanějšímu písmenu v českých textech. Dojdeme k závěru, že klíč je $k_1 = 10, k_2 = 0, k_3 = 19, k_4 = 25$, tj. heslo `katz`*. Dešifrovaný text (pochází z 3. dílu Haškova Švejka):*

```
systemkteryjsemvamvysvetlovaljenejenjedenznejlepsichalemuzem
erictnedostiznychvsechnaoddeleniproprotispionaznasichneprate
lskychstabumohoujitinahrneckdybyserozkrajelineprectounasesif
ryjesttonecozcelanovehotytosifrynemajipredchudce
```

Tabulka 2.4: Frekvenční analýza podposloupností šifrového textu

pozice $1, 5, \ldots$	pozice $2, 6, \ldots$	pozice $3, 7, \ldots$	pozice $4, 8, \ldots$
COOCKCVVOTXT	YMREMVOJJEZL	LDRFOXOXXWGX	SSIUXSZMMDDO
CKEODYJRMYVZ	ILZRNSNVHDER	VXXBXMRLGWGH	GLLBCHBDZDHO
BCXKRBVMKYTX	OPASNASHBHIA	MBSBXMDLNHMA	HNMBODXSLTHQ
XNCJTXOEOBCX	EYEKEECNSYTE	VUKKEIMTBCMV	JXNZHQNREDNN
JKOISXTOE	CNHTFEIDD	XHHHKFIVV	KUSRXZQGD

Relativní frekvence v procentech

A	$0,00$	A	$5,26$	A	$1,75$	A	$0,00$
B	$5,26$	B	$1,75$	B	$7,02$	B	$5,26$
C	$12,28$	C	$3,51$	C	$1,75$	C	$1,75$
D	$1,75$	D	$5,26$	D	$3,51$	D	$12,28$
E	$5,26$	E	$15,79$	E	$1,75$	E	$1,75$
F	$0,00$	F	$1,75$	F	$3,51$	F	$0,00$
G	$0,00$	G	$0,00$	G	$5,26$	G	$3,51$
H	$0,00$	H	$7,02$	H	$8,77$	H	$8,77$
I	$1,75$	I	$5,26$	I	$3,51$	I	$1,75$
J	$5,26$	J	$3,51$	J	$0,00$	J	$1,75$
K	$8,77$	K	$1,75$	K	$5,26$	K	$1,75$
L	$0,00$	L	$3,51$	L	$5,26$	L	$5,26$
M	$3,51$	M	$3,51$	M	$10,53$	M	$5,26$
N	$1,75$	N	$8,77$	N	$1,75$	N	$8,77$
O	$14,04$	O	$3,51$	O	$3,51$	O	$5,26$
P	$0,00$	P	$1,75$	P	$0,00$	P	$0,00$
Q	$0,00$	Q	$0,00$	Q	$0,00$	Q	$5,26$
R	$3,51$	R	$5,26$	R	$3,51$	R	$3,51$
S	$1,75$	S	$7,02$	S	$1,75$	S	$8,77$
T	$8,77$	T	$3,51$	T	$1,75$	T	$1,75$
U	$0,00$	U	$0,00$	U	$1,75$	U	$3,51$
V	$7,02$	V	$3,51$	V	$8,77$	V	$0,00$
W	$0,00$	W	$0,00$	W	$3,51$	W	$0,00$
X	$12,28$	X	$0,00$	X	$15,79$	X	$7,02$
Y	$5,26$	Y	$5,26$	Y	$0,00$	Y	$0,00$
Z	$1,75$	Z	$3,51$	Z	$0,00$	Z	$7,02$

Tabulka 2.5: Frekvence písmen v českých textech (bez diakritiky a mezery)

Znak	*frekvence (%)*	*Znak*	*frekvence (%)*
a	8,85	n	6,49
b	1,65	o	8,15
c	3,19	p	3,19
d	3,66	q	0,00
e	10,63	r	4,49
f	0,24	s	5,55
g	0,24	t	5,43
h	2,36	u	4,13
i	6,97	v	4,60
j	2,60	w	0,00
k	3,90	x	0,12
l	4,01	y	2,83
m	3,42	z	3,31

2.1.4 Vernamova šifra

Význačné postavení mezi symetrickými kryptosystémy má *Vernamova šifra (1917)*. Je založena na principu Vigenèrovy šifry s tím, že klíč (heslo) má stejnou délku jako zpráva, např.

- text knihy, na které se odesílatel a příjemce dohodnou[3],
- zcela náhodná posloupnost písmen (pro každou zprávu nová).

V tomto druhém pojetí je Vernamova šifra označována jako *one-time pad* a je dosud používána pro předávání výjimečně tajných zpráv. Lze totiž dokázat, že se jedná o tzv. *perfektní kryptosystém*, tj. ze znalosti šifrového textu se nic nedozvíme o zprávě (ke každému otevřenému textu najdeme klíč, který jej kóduje do libovolného šifrového textu). Předem připravený klíč potřebné délky se po použití k šifrování (u odesílatele) a dešifrování (u adresáta) zničí, aby nemohlo dojít k dešifrování odposlechnutého šifrového textu ani v budoucnu.

Příklad 2.7 *Při použití Vernamovy šifry vidíme, že ke stejnému šifrovému textu můžeme dojít zašifrováním libovolného otevřeného textu vhodným*

[3]Úskalí šifer založených na textu knihy jsou popsána např. ve 3. díle Haškova Švejka.

klíčem:

Otevřené texty:	`onetimepad`	`greenfluid`
Klíče:	`tbfrgfarfm`	`bxfgbmtmxm`
Šifrové texty:	`IPKLPSFHGQ`	`IPKLPSFHGQ`

V případě binární abecedy dostáváme z principu Vernamovy šifry jednoduché (ale účinné) šifrování založené na operaci $\oplus$ (sčítání modulo 2):

Zpráva: $M = m_1 \dots m_n$ je bitový řetězec délky n, $m_i \in \{0, 1\}$.

Klíč: $K = k_1 \dots k_n$ je bitový řetězec stejné délky n, přičemž $k_i \in \{0, 1\}$ jsou předem náhodně generované.

Šifra: $C = M \oplus K$ je součtem po bitech: $c_i = m_i \oplus k_i$.

Dešifrování: $M = C \oplus K$, neboť

$$c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i \oplus (k_i \oplus k_i) = m_i.$$

2.1.5 Enigma

V 1. pol. minulého století dosáhly velkého rozšíření elektromechanické realizace *složených* substitučních šifer – rotační šifrové stroje, např. slavná Enigma (viz např.[17]). Stroj byl tvořen soustavou propojených disků, které se při šifrování po každém zašifrovaném písmenu pootočí (jako v mechanickém počitadle u tachometru). Další písmeno je tedy šifrováno jinou substitucí. Každý disk realizuje jednu pevnou permutaci latinské abecedy, která je nastavením disku cyklicky posunována (jako u Césarovy šifry). Výsledná substituce vzniká pro každou pozici v textu složením těchto permutací posunutých podle aktuálního nastavení disků. Šifrovacím klíčem je zde výchozí nastavení disků (u tří disků jde o $6.26.26.26 = 105456$ možností) a některých dalších parametrů stroje. Na rozdíl od Vigenèrovy šifry, kde se stejná substituce opakuje po d písmenech, v základním typu stroje Enigma se třemi disky vznikne stejná substituce až po 16900 písmenech.

Rozluštění šifry Enigma bylo velkým úspěchem polských a britských kryptoanalytiků (jmenujme alespoň M.Rejewského a A.Turinga) a přispělo významně k vítězství spojenců ve druhé světové válce. Vyvinuté kryptoanalytické postupy vedoucí k prolomení kryptosystému Enigma nelze na malém prostoru jednoduše popsat; poznamenejme, že potřeby jejich realizace vedly (podle [17]) ke konstrukci prvních elektromechanických a elektronkových výpočetních zařízení založených na myšlence univerzálního Turingova stroje.

2.2 Šifrový standard DES

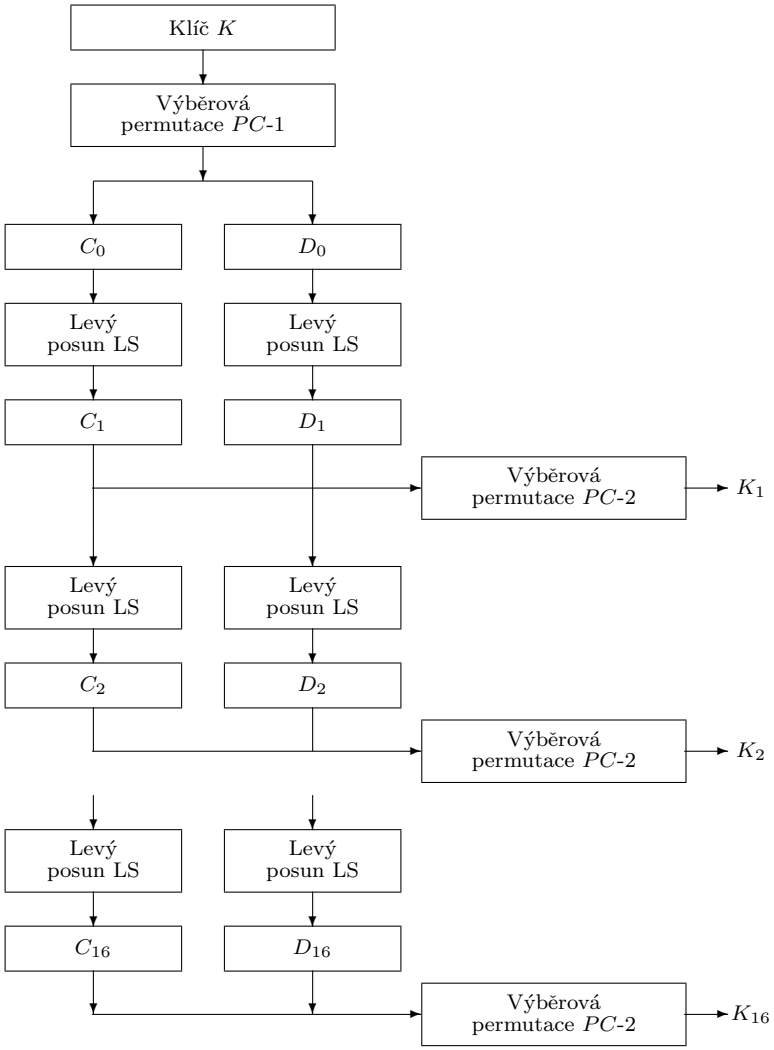
Kryptoanalytické metody uvedené v předcházejících částech byly založeny na tom, že jednoduché šifrovací systémy (Césarova i Vigenèrova šifra) vytvářejí šifrový text, z jehož statistických vlastností lze dělat závěry týkající se šifrovaného textu. Proto je cílem pokročilých šifrovacích algoritmů transformovat otevřený text rovnoměrně do šifrového textu tak, aby byly šifrové texty stejně pravděpodobné. Teoreticky tuto myšlenku formuloval již C. Shannon a dále ji rozpracoval Feistel při vytváření kryptosystému LUCIFER firmy IBM. Na základě tohoto kryptosystému byl vyvinut a v roce 1977 zaveden v USA šifrový standard DES (Data Encryption Standard), který se stal na dvě desetiletí nejrozšířenějším symetrickým kryptosystémem určeným pro digitální přenos dat. Na algoritmu DES ukážeme základní rysy tzv. *smíšených substitučně-transpozičních šifer*. V další části pak stručně popíšeme nový americký šifrový standard AES, který DES nahrazuje.

DES slouží k šifrování bloků 64 bitů (utajovanou zprávu ve formě bitového řetězce je nutno předem rozdělit na 64-bitové bloky a poslední blok doplnit na 64 bitů). Šifrovací klíč je určen 56 bity – existuje tedy celkem 2^{56} možných klíčů. Algoritmus DES provádí postupně 16 iterací šifrovacího schématu tvořeného pevně danými substitucemi, transpozicemi a aplikací v každém iteračním kroku jiného 48-bitového klíče, který je odvozován ze zadaného 56-bitového klíče. Proto je před vlastním šifrováním nutno připravit ze zadaného (56-bitového) klíče K šestnáct 48-bitových klíčů $K_1, \ldots, K_{16}$. Při vlastní aplikaci klíče se používá bitová operace $\oplus$ (součet modulo 2).

V dalším popíšeme zvlášť postup odvozování 48-bitových klíčů (tento postup je obvykle nazýván *expanzí klíčů*) a vlastní šifrovací funkci.

2.2.1 Expanze klíčů

56-bitový klíč K je zadáván jako 64 bitů (8 bytů), bity č. $8, 16, \ldots, 64$ slouží jako paritní kontrola. Z klíče je odvozeno 16 klíčů $K_1, \ldots, K_{16}$, které jsou postupně využívány při šifrování.



Obrázek 2.3: Postup odvozování klíčů v DES.

Postup odvození klíčů K_i (viz obrázek 2.3) je následující: nejdříve se aplikuje *výběrová permutace* PC-1 – vynechání paritních bitů a transpozice ostatních dle tabulky 2.6. Výsledkem je 56-bitový řetězec $PC\text{-}1(K)$ začínající 57. bitem klíče K, za ním následuje 49. bit klíče K a tak dále, na konci je 4. bit klíče K.

Tabulka 2.6: Výběrová permutace klíče PC-1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Tabulka 2.7: Velikost levých posunů LS

Iterace	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Posun	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Tento 56-bitový řetězec je rozdělen na poloviny C_0, D_0 po 28 bitech: $PC\text{-}1(K) = C_0D_0$. Pak je v krocích $i = 1, \ldots, 16$ prováděna v každé polovině určená transpozice LS_i – daný *cyklický posun* pozic vlevo (viz tabulka 2.7): $C_i = LS_i(C_{i-1}), D_i = LS_i(D_{i-1})$. Výsledek: $K_i = PC\text{-}2(C_iD_i)$, kde PC-2 (viz tab. 2.8) je *výstupní výběrová permutace*, při níž proběhne též zúžení na 48 bitů (není použit např. bit č. 9 řetězce C_iD_i).

Příklad 2.8 *Jako příklad uvedeme odvození prvních tří klíčů ze zadaného klíče K:*
`1001001100110100010101110111100110011011101111001101111111110001`
Výsledek výběrové permutace PC-1(K):
`1100110000000000011001100111111111111000010101010111100001010101`

Průběh levých posunů LS:

Tabulka 2.8: Permutace klíče PC-2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

$C_0 =$ 11100011100110010101011111
$D_0 =$ 10101010110011001111100011110
$C_1 =$ 11110101010100110011100011111
$D_1 =$ 11110001111001100110101010101
$C_2 =$ 11000111001100101010111111
$D_2 =$ 01010101100110011111000111101
$C_3 =$ 11110101010100110011100011111
$D_3 =$ 11110001111001100110101010101
$C_4 =$ 11000111001100101010111111
$D_4 =$ 01010101100110011111000111101
$C_5 =$ 11110101010100110011100011111
$D_5 =$ 11110001111001100110101010101
$C_6 =$ 11000111001100101010111111
$D_6 =$ 01010101100110011111000111101
$C_7 =$ 11110101010100110011100011111
$D_7 =$ 11110001111001100110101010101
$C_8 =$ 11100011100110010101011111
$D_8 =$ 10101010110011001111100011110
$C_9 =$ 11101010101001100111000111111
$D_9 =$ 11100011110011001101010101
$C_{10} =$ 11100011100110010101011111
$D_{10} =$ 10101010110011001111100011110
$C_{11} =$ 11101010101001100111000111111
$D_{11} =$ 11100011110011001101010101
$C_{12} =$ 11100011100110010101011111
$D_{12} =$ 10101010110011001111100011110
$C_{13} =$ 11101010101001100111000111111
$D_{13} =$ 11100011110011001101010101
$C_{14} =$ 11100011100110010101011111
$D_{14} =$ 10101010110011001111100011110
$C_{15} =$ 11110101010100110011100011111
$D_{15} =$ 11110001111001100110101010101

Odvozené klíče (výsledky druhé výběrové permutace):

$K_1 = PC\text{-}2(C_1D_1)$:
000110110000001011111111111111000111000001110010
$K_2 = PC\text{-}2(C_2D_2)$:
011110011110111011011001110110111100100111100101
$K_3 = PC\text{-}2(C_3D_3)$:
010101011111111011000101001000010110011111001001
$K_4 = PC\text{-}2(C_4D_4)$:
011100101010110111010111110110110011010100011101
$K_5 = PC\text{-}2(C_5D_5)$:
011111011110110000000111111010110101001110101000
$K_6 = PC\text{-}2(C_6D_6)$:
011000111010010110111110010100000111101100101111
$K_7 = PC\text{-}2(C_7D_7)$:
111111001000010010110111111101100001100010111100
$K_8 = PC\text{-}2(C_8D_8)$:
111101111000101000111010110000010011101111111011
$K_9 = PC\text{-}2(C_9D_9)$:
111000001111101111101011111011011110011110000001
$K_{10} = PC\text{-}2(C_{10}D_{10})$:
101100011111011101000111101110100100011001001111
$K_{11} = PC\text{-}2(C_{11}D_{11})$:
011000010101111111010011110111101101001110000110
$K_{12} = PC\text{-}2(C_{12}D_{12})$:
011101011111000111110101100101000110011111101001
$K_{13} = PC\text{-}2(C_{13}D_{13})$:
100101111100010111010011111111010101110100100001
$K_{14} = PC\text{-}2(C_{14}D_{14})$:
011111110100001110110111111100101110011100111010
$K_{15} = PC\text{-}2(C_{15}D_{15})$:
101111111001000110001101001111010011111100001010
$K_{16} = PC\text{-}2(C_{16}D_{16})$:
110010110011110111001011000011100001011111110101

Tabulka 2.9: Vstupní Permutace IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

2.2.2 Algoritmus šifrování

Schéma šifrování je zachyceno na obr. 2.4. Vstupem je šifrovaný 64-bitový blok T a samozřejmě 16 připravených 48-bitových klíčů $K_1, \ldots, K_{16}$. Nejprve proběhne *vstupní permutace* IP (viz tab. 2.9), jejímž výsledkem je 64-bitový blok $T_0 = IP(T)$ začínající 58. bitem bloku T, pak následuje 50. bit bloku T a tak dále; na konci je 7. bit bloku T. Výsledný blok T_0 je poté rozdělen na levou část L_0 (prvních 32 bitů) a pravou část R_0 (zbývajících 32 bitů).

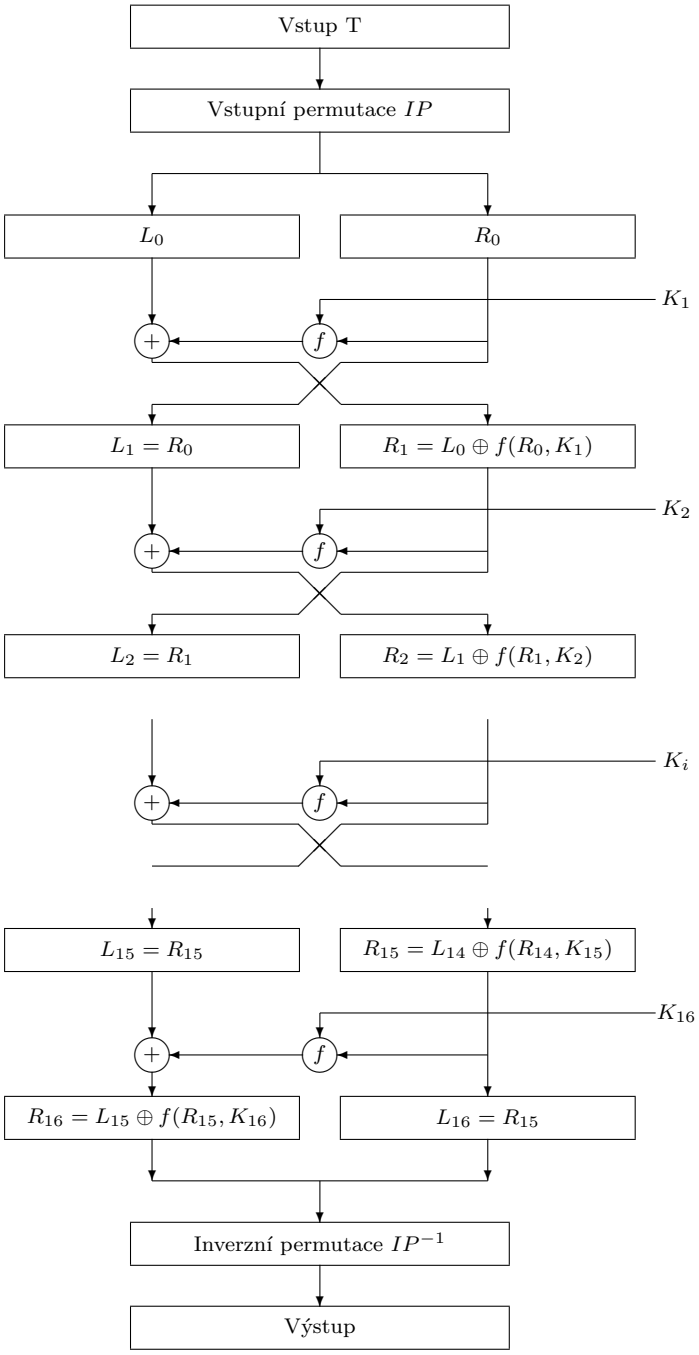
Celkově je odvození klíčů tvořeno složením permutací a cyklických posunů – každý výsledný klíč je tedy dán výběrem a přeskupením jistých 48 bitů v klíči K. Pořadí vybraných bitů pro každý odvozený klíč udává tabulka 2.8.

Vlastní šifrování se skládá z *16 iterací* (rund). Pro $i = 1, \ldots, 16$ je vstupem i-té iterace výsledek předchozí iterace T_{i-1} tvořený levou částí L_{i-1} a pravou částí R_{i-1}. Výstupem i-té operace je 64-bitový blok T_i, jehož levá část L_i a pravá část R_i jsou určeny takto:

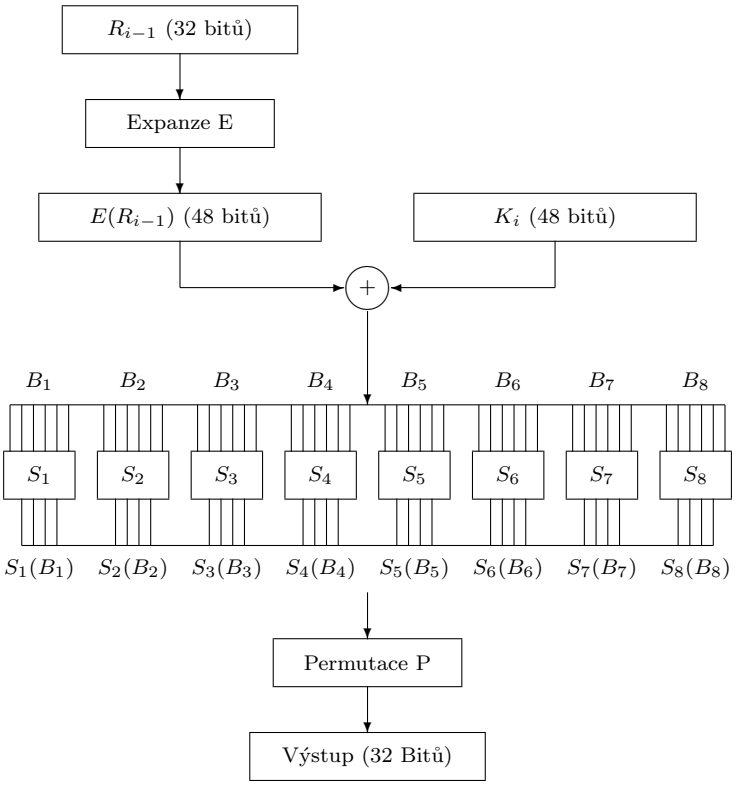
$L_i = R_{i-1}$ ... přemístění pravé poloviny doleva,
$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$ ... výpočet pravé poloviny – použití šifrovací funkce f s odvozeným klíčem K_i.

Šifrovací funkce f (viz obr. 2.5) realizuje vlastní aplikaci klíče: Pravá polovina R_{i-1} (32 bitů) je nejdříve rozšířena podle tabulky expanze (viz levá část tabulky 2.10) na 48-bitový řetězec $E(R_{i-1})$. Prvních šest bitů je nyní tvořeno 32., 1., 2., 3., 4. a 5. bitem řetězce R_{i-1}, ... a tak dále; posledních šest bitů tvoří 28., 29., 30., 31., 32. a 1. bit řetězce R_{i-1}. K výslednému 48-bitovému řetězci $E(R_{i-1})$ je přičten modulo 2 odvozený klíč K_i.

Výsledek $E(R_{i-1}) \oplus K_i$ je rozdělen na 6-bitové řetězce $B_1, \ldots, B_8$ a transformován substitučními tabulkami, tzv. *S-boxy* (viz tab. 2.11, kde číslo řádku tabulky S-boxu S_j je dáno binárně 1. a 6. bitem řetězce B_j, číslo sloupce



Obrázek 2.4: Šifrovací schéma DES.



Obrázek 2.5: Schéma šifrovací funkce f.

2. – 5. bitem). Výstupy S-boxů jsou čtyřbitové řetězce $S_1(B_1), \ldots, S_8(B_8)$. Jejich spojením vznikne 32-bitový řetězec, který je transformován nakonec *permutací* P (viz pravá část tabulky 2.10):

$$f(R_{i-1}, K_i) = P(S_1(B_1) \ldots S_8(B_8)).$$

Výsledkem 16 iterací šifrování (rund) je blok $R_{16}L_{16}$, který je na závěr transformován pomocí *inverzní vstupní permutace* IP^{-1} (viz tab. 2.12) na výstupní šifrový text $IP^{-1}(R_{16}L_{16})$.

Při dešifrování se používá stejný algoritmus jako u šifrování, ale odvozené klíče jsou aplikovány v pořadí $K_{16}, K_{15}, \ldots, K_1$, neboť platí $R_{i-1} = L_i$ a $L_{i-1} = R_i \oplus f(L_i, K_i)$.

Tabulka 2.10: Tabulky expanze a permutace

Expanze

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Permutace P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Tabulka 2.11: S-boxy

		Sloupec															
Box	řádek	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Tabulka 2.12: Výstupní permutace IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Příklad 2.9 *V příkladu uvedeme podrobně průběh všech 16 iterací DES.*
Šifrujeme otevřený text T:
T: 0000000100100011010001010110011111000100110101011111001110111101111
Používáme klíč K:
K: 1001001100110100010101111011110011001101110111110011011111111110001
Dostáváme blok $T_0 = L_0R_0$ jako výsledek vstupní permutace $IP(T)$:
L_0 = 11001100000000001100110011111111
R_0 = 11110000101010101111000010101010

A dále

$$
\begin{aligned}
E(R_0) &= 011110100001010101010101011110100001010101010101\\
K_1 &= 000110110000000101111111111111110001110000011100 10\\
E(R_0)\oplus K_1 &= 011000010001011110101010100001100110010100100111\\
S\text{ – box výstup} &= 01011100100010111011010110010111\\
f(R_0,K_1) &= 10100011010010101010100111111011\\
L_2 = R_1 &= 01101111010010100110010100000100\\
E(R_1) &= 001101011110101001010100001100001010100000001000\\
K_2 &= 011110011110111011011001110110111100100111100101\\
E(R_1)\oplus K_2 &= 010011000000010010001101111010110110000111101101\\
S\text{ – box výstup} &= 01101111110100000011101001111000\\
f(R_1,K_2) &= 01111110001110011100011110010000\\
L_3 = R_2 &= 10001110100100110011011100111010\\
E(R_2) &= 010001011101010010100110100110101110100111110101\\
K_3 &= 010101011111111011000101001000010110011111 10011001\\
E(R_2)\oplus K_3 &= 000100000010100100101100110110000010011001101100\\
S\text{ – box výstup} &= 11010001010001110101000100101110\\
f(R_2,K_3) &= 10101000110001111111010000100010\\
L_4 = R_3 &= 11000111100011011001000100100110\\
E(R_3) &= 011000001111110001011011110010100010100100001101\\
K_4 &= 011100101010110111010111110110110011010100011101\\
E(R_3)\oplus K_4 &= 000100100101000110001100000100010001110000010000\\
S\text{ – box výstup} &= 11011010111010010100011010101010\\
f(R_3,K_4) &= 11001000101011111000010101001111\\
L_5 = R_4 &= 01000110001111001011001001110101\\
E(R_4) &= 101000001100000111111001010110100100001110101010\\
K_5 &= 011111011110110000000111111010110101001110101000\\
E(R_4)\oplus K_5 &= 110111010010110111111110101100010001000000000010\\
S\text{ – box výstup} &= 11100111001101000111011001000010\\
f(R_4,K_5) &= 01100100101101101101001010011100\\
L_6 = R_5 &= 10100011001110110100001110111010
\end{aligned}
$$

$$
\begin{aligned}
E(R_5) &= 010100000110100111110110101000000111110111110101\\
K_6 &= 011000111010010110111110010100000111101100101111\\
E(R_5) \oplus K_6 &= 001100111100110001001000111100000000011011011010\\
S\text{ – box výstup} &= 10110010010000000000110011110000\\
f(R_5, K_6) &= 01010010100100010000011000001011\\
L_7 = R_6 &= 00010100101011011011010001111110\\
E(R_6) &= 000010101001010101011011110110101000001111111100\\
K_7 &= 111111001000010010110111111101100001100010111100\\
E(R_6) \oplus K_7 &= 111101100001000111101100001011001001101101000000\\
S\text{ – box výstup} &= 01101101100101110111011110101101\\
f(R_6, K_7) &= 10101100011011001111111110111001\\
L_8 = R_7 &= 00001111010101111011110000000011\\
E(R_7) &= 100001011110101010101111110111111000000000000110\\
K_8 &= 111101111000101000111010110000010011101111111011\\
E(R_7) \oplus K_8 &= 011100100110000010010101000111101011101111111101\\
S\text{ – box výstup} &= 00001011000000101100010101110110\\
f(R_7, K_8) &= 01000011010111100110010000101000\\
L_9 = R_8 &= 01010111111100111101000001010110\\
E(R_8) &= 001010101111111110100111111010100000001010101100\\
K_9 &= 111000001111101111010111110110111100111110000001\\
E(R_8) \oplus K_9 &= 110010100000010001001100000001111110010100101101\\
S\text{ – box výstup} &= 11000000001010011110011010011000\\
f(R_8, K_9) &= 10001011101001001000000011001101\\
L_{10} = R_9 &= 10000100111100110011110011001110\\
E(R_9) &= 010000001001011110100110100111111001011001011101\\
K_{10} &= 101100011111011101000111101110100100011001001111\\
E(R_9) \oplus K_{10} &= 111100010110000011100001001001011101000000010010\\
S\text{ – box výstup} &= 01011101011100110100001101001001\\
f(R_9, K_{10}) &= 10001100011111011110100000010110\\
L_{11} = R_{10} &= 11011011100011100011100001000000\\
E(R_{10}) &= 011011110111110001011100000111110000001000000001\\
K_{11} &= 011000010101111111010011110111101101001110000110\\
E(R_{10}) \oplus K_{11} &= 000011100010001110001111110000011101000110000111\\
S\text{ – box výstup} &= 11111110010100111111001111101000\\
f(R_{10}, K_{11}) &= 11101101111111011010011010010011\\
L_{12} = R_{11} &= 01101001000011101001101001011101\\
E(R_{11}) &= 101101010010100001011101010011111010000101111101 0\\
K_{12} &= 011101011111100011111010110010100011001111110100 1\\
E(R_{11}) \oplus K_{12} &= 110000001101100110101000110110110010010100010011\\
S\text{ – box výstup} &= 11111000100111000101000010010101\\
f(R_{11}, K_{12}) &= 00100110100011001001101101100011\\
L_{13} = R_{12} &= 11111101000000101010001100100011
\end{aligned}
$$

$E(R_{12}) =$ 111111111010100000000101010100000110100100000111
$K_{13} =$ 100101111100010111010011111110101011101001000001
$E(R_{12}) \oplus K_{13} =$ 011010000110110111010110101010101101001101000110
S – *box výstup* $=$ 10011110001101011101111100010100
$f(R_{12}, K_{13}) =$ 11110111101011000011000000111110
$L_{14} = R_{13} =$ 10011110101000101010101001100011
$E(R_{13}) =$ 110011111101010100000101010101010100001100000111
$K_{14} =$ 011111110100001110110111111100101110011100111010
$E(R_{13}) \oplus K_{14} =$ 101100001001011010110010101001111010010000111101
S – *box výstup* $=$ 00101111010000010001110100110110
$f(R_{13}, K_{14}) =$ 11110010000010110110011000111000
$L_{15} = R_{14} =$ 00001111000010011100010100011011
$E(R_{14}) =$ 100001011110100001010011111000001010100011110110
$K_{15} =$ 101111111001000110001101001111010011111100001010
$E(R_{14}) \oplus K_{15} =$ 001110100111100111011110110111011001011111111100
S – *box výstup* $=$ 10000001000011111001000001100101
$f(R_{14}, K_{15}) =$ 10100001110100000101110001100000
$L_{16} = R_{15} =$ 00111111011100101111011000000011
$E(R_{15}) =$ 100111111110101110100101011110101100000000000110
$K_{16} =$ 110010110011110111001011000011100001011111110101
$E(R_{15}) \oplus K_{16} =$ 010101001101011001101110011101001101011111110011
S – *box výstup* $=$ 11001000110011011000100101101100
$f(R_{15}, K_{16}) =$ 10011001100110011011010101100000
$R_{16} =$ 10010110100100000111000001111011

A konečně výsledný šifrový text $IP^{-1}(R_{16}L_{16})$ *je:*
10000011111010111100100010000001111111011010110100101101010110000.

2.2.3 Vlastnosti DES

Šifrový standard DES představuje blokově-substituční kryptosystém – pro daný klíč je každý z 2^{64} možných 64-bitových bloků substituován šifrovým 64-bitovým blokem. Tato substituce samozřejmě nemůže být zadána tabulkou, ale je ke každému bloku zprávy určována popsaným algoritmem šifrování.

Statisticky byly při ověřování DES testovány zejména vlastnosti *konfúze* a *difúze*, tj. komplikovaný vliv každého bitu otevřeného textu a klíče na každý bit šifrového textu. Bylo statisticky potvrzeno, že změna jednoho bitu v otevřeném textu vede k 50% pravděpodobnosti změny každého bitu šifrového textu a že není korelace mezi otevřeným a šifrovým textem, ani mezi šifrovým textem a klíčem. Významné jsou tzv. nelineární vlastnosti substitučních S-boxů – tj. každý výstupní bit S-boxu je nelineární funkcí všech vstupních bitů.

Je zajímavé, že tvůrcům DES zřejmě unikla vlastnost *komplementarity*, která snižuje prostor klíčů při luštění na polovinu. Označíme-li $x' = 1 - x$ doplněk bitu x a X' doplněk bitového řetězce X, pak z $C = E_K(M)$ plyne $C' = E_{K'}(M')$.

2.2.4 Útoky proti DES

Hlavní kritika DES se od počátku týkala příliš krátkého klíče (56 bitů, tj. 2^{56} možných klíčů). Diffie a Hellman (Stanford University) už v roce 1975 provedli odhady možností rozluštění klíče rychle se vyvíjející počítačovou technologií, které se později potvrdily v plném rozsahu v rámci vyhlášených "DES Challenge" v letech 1997-99. Společnost RSA Data Security při nich zadala šifrový text, který se zájemci mohli pokoušet vyluštit.

V první soutěži zvítězil Verser tím, že rozdělil klíčový prostor dobrovolníkům, kteří se prostřednictvím Internetu podíleli svými počítači na prolomení šifry. Na začátku akce v březnu 1997 bylo tak zapojeno 20 počítačů, v červnu 1997 až 14 000 počítačů najednou. Celkově bylo vyzkoušeno 72 057 594 037 927 936 klíčů a nalezen správný klíč:

1000010101011000100010010001101010110000110010000101000110110110

(na jednom z připojených počítačů s procesorem Pentium 90 MHz a rychlostí řešení 250 000 klíčů/s). Utajená zpráva zněla:

`Strong cryptography makes the world a safer place.`

V dalším roce zvítězil speciálně vyrobený "luštící stroj" DES-cracker (společnost Electronic Frontier Foundation) obsahující 24 desek se 64 čipy po 24 minijednotkách. Minijednotka zkouší všechny kombinace 32 dolních bitů pro zadaných 24 horních bitů (56bitového klíče) a dané pozice šifrového textu (např. 1–8). Bylo dosaženo rychlosti, při které minijednotka testovala 2.5 mil. klíčů/s, tj. deska 3.84 miliardy klíčů/s, takže celý klíčový prostor mohl být prohledán asi za 9 dní.

V roce 1999 již dokázala kombinace Superpočítače Deep Crack a 100 000 PC prohledat 245 miliard klíčů za sekundu, a nalézt správný klíč za 22 h 15 min.

Obavy z útoků proti DES vedly k úvahám o vícenásobném šifrování. Je zajímavé, že prosté dvojnásobné šifrování klíči K_1, K_2 ($C = E_{K_2}(E_{K_1}(M))$) vede pouze ke zdvojnásobení prohledávacího prostoru klíčů. Jako účinný se ukázal teprve postup trojnásobné aplikace DES se dvěma klíči označovaný za Triple DES, u nějž vzroste prohledávací prostor z 2^{56} na 2^{112} možností. Při něm je se zadanými K_1, K_2 dvakrát aplikován postup šifrování a jednou dešifrovací postup:

$$C = E_{K_1}(D_{K_2}(E_{K_1}(M))).$$

2.2.5 Diferenciální kryptoanalýza DES

V tomto odstavci popíšeme na ukázku zajímavou kryptoanalytickou metodu, vyvinutou E. Bihamem a A. Shamirem – 1990. Slovo "'diferenciální"' v jejím názvu odkazuje na fakt, že binární součet mod2 je zároveň totéž co rozdíl (diference) bitů.

Diferenciální kryptoanalýza DES je založena na rozpoznání, že při výpočtu šifrových funkcí $f(R_a, K)$ a $f(R_b, K)$ se vliv klíče na diferenci vstupů $R_a \oplus R_b$ eliminuje. Diferenci šifer lze totiž určit z diference vstupů:

$$\begin{gathered}(E(R_a) \oplus K) \oplus (E(R_b) \oplus K) = \\ = (E(R_a) \oplus E(R_b)) \oplus (K \oplus K) = E(R_a) \oplus E(R_b) = \\ = E(R_a \oplus R_b).\end{gathered}$$

Prvním krokem metody je apriorní diferenciální analýza všech S-boxů S_j:

Uvažujme řetězce $\Delta \in \{0,1\}^6$ a $\Gamma \in \{0,1\}^4$ a označme

$$IN_j(\Delta, \Gamma) = \{B \in \{0,1\}^6; S_j(B) \oplus S_j(B \oplus \Delta) = \Gamma\}.$$

Množina $IN_j(\Delta, \Gamma)$ obsahuje možné vstupy do S-boxu S_j, které respektují relaci mezi vstupní diferencí Δ a výstupní diferencí Γ.

Příklad 2.10 *Diferenciální analýza S-boxů ukazuje značné rozdíly ve velikosti množiny $IN_j(\Delta, \Gamma)$. V tabulce je část výsledků analýzy S-boxu S_1 pro $\Delta = 110100$.*

Γ	$IN_j(110100, \Gamma)$
0000	
0001	000011, 001111, 011110, 011111, 101010, 101011, 110111, 111011
0010	000100, 000101, 001110, 010001, 010010, 010100, 011010, 011011, 100000, 100101, 010110, 101110, 101111, 110000, 110001, 111010
0011	000001, 000010, 010101, 100001, 110101, 110110
0101	
0110	
0111	000000, 001000, 001101, 010111, 011000, 011101, 100011, 101001, 101100, 110100, 111001, 111100
1000	001001, 001100, 011001, 101101, 111000, 111101
1001	
1010	
1011	
1100	
1101	000110, 010000, 010110, 011100, 100010, 100100, 101000, 110010
1110	
1111	000111, 001010, 001011, 110011, 111110, 111111

Označíme-li nyní pro libovolné dva vstupy $B, B^* \in \{0,1\}^6$:

$$\begin{aligned} B' &= B \oplus B^* && \text{– diference vstupů,} \\ C' &= S_j(B) \oplus S_j(B^*) && \text{– diference výstupů,} \end{aligned}$$

pak platí $B, B^* \in IN_j(B', C')$, neboť

$$S_j(B) \oplus S_j(B \oplus B') = S_j(B) \oplus S_j(B \oplus (B \oplus B^*)) = S_j(B) \oplus S_j(B^*) = C'.$$

Nyní již můžeme přikročit k testování klíče použitého na vstup do S-boxů.

Šifrovací funkce f má tvar $f(R_{i-1}, K_i) = P(S_1(B_1) \dots S_8(B_8))$,
kde $B = B_1 B_2 \dots B_8 = E(R_{i-1}) \oplus K_i$.
Označme dále: $E = E_1 E_2 \dots E_8 = E(R_{i-1})$,
$J = J_1 J_2 \dots J_8 = K_i$.

Pro vstup do S-boxu $S_j (j = 1, \ldots, 8)$ platí $B_j = E_j \oplus J_j$, a tedy příslušnou část klíče můžeme určit jako $B_j \oplus E_j = J_j$. Pro diferenci B'_j dvou vstupů B_j, B^*_j přitom platí, že je rovna diferenci E'_j příslušných vstupů E_j, E^*_j před přičtením klíče:

$$B'_j = B_j \oplus B^*_j = (E_j \oplus J_j) \oplus (E^*_j \oplus J_j) = E_j \oplus E^*_j = E'_j.$$

Označíme-li

$$C'_j = S_j(B_j) \oplus S_j(B^*_j),$$

pak pro možné vstupy B_j platí

$$B_j \in IN_j(B'_j, C'_j) = IN_j(E'_j, C'_j)$$

a tedy lze určit množinu řetězců J_j, které mohly být použity jako klíč k šifrování před vstupem do S_j:

$$J_j \in \mathrm{Test}_j(E_j, E^*_j, C'_j) = \{B_j \oplus E_j; B_j \in IN_j(E'_j, C'_j)\}.$$

Množiny $\mathrm{Test}_j(E_j, E^*_j, C'_j)$ budeme nazývat testové množiny. Použitý klíč budeme hledat v průniku testových množin pro různé vstupní a výstupní dvojice.

Příklad 2.11 *Určení testové množiny možných klíčů uvedeme pro S-box S_1, vstupy $E_1 = 000001, E^*_1 = 110101$ a diferenci výstupů $C'_1 = 1101$. Diference vstupů je $E'_1 = E_1 \oplus E^*_1 = 110100$. Podle tabulky z Příkladu 2.10 dostáváme*

$$IN_1(110100, 1101) = \{000110, 010000, 010110, 011100, 100010, 100100, 101000, 110010\}.$$

Použitá část klíče J_1 tedy musí být prvkem testové množiny

$$Test_1(000001, 110101, 1101) = \{000111, 010001, 010111, 011101, 100011, 100101, 101001, 110011\}.$$

Takto připravené prostředky diferenciální kryptoanalýzy nyní využijeme k útoku se zadávanými otevřenými texty na "3-rundový" DES (tj. v situaci, kdy můžeme získat výsledek šifrování po 3 iteracích). Zadáváme dvojice otevřených textů takové, že (po vstupní permutaci, která nemá kryptoanalytický význam) mají shodné pravé poloviny:

$$T = L_0 R_0, \qquad T^* = L^*_0 R^*_0, \qquad R_0 = R^*_0$$

a získáváme k nim šifrové texty po 3 iteracích šifrovacího algoritmu DES:

$$T_3 = L_3 R_3, \qquad T^*_3 = L^*_3 R^*_3.$$

Rozborem jednotlivých iterací zjistíme, že pro pravé strany nyní platí

$$\begin{aligned} R_3 = L_2 \oplus f(R_2, K_3) = R_1 \oplus f(R_2, K_3) = \\ = L_0 \oplus f(R_0, K_1) \oplus f(R_2, K_3), \\ \text{analogicky} \qquad R_3^* = L_0^* \oplus f(R_0^*, K_1) \oplus f(R_2^*, K_3). \end{aligned}$$

Odtud plyne (vzhledem k tomu, že $R_0 = R_0^*$):

$$\begin{aligned} &R_3' = R_3 \oplus R_3^* = \\ &= (L_0 \oplus f(R_0, K_1) \oplus f(R_2, K_3)) \oplus (L_0^* \oplus f(R_0^*, K_1) \oplus f(R_2^*, K_3)) = \\ &= (L_0 \oplus L_0^*) \oplus (f(R_0, K_1) \oplus f(R_0^*, K_1)) \oplus (f(R_2, K_3) \oplus f(R_2^*, K_3)) = \\ &= L_0' \oplus f(R_2, K_3) \oplus f(R_2^*, K_3). \end{aligned}$$

To ale znamená, že

$$f(R_2, K_3) \oplus f(R_2^*, K_3) = R_3' \oplus L_0',$$

a tedy pro diferenci výstupů z S-boxů platí

$$C' = C \oplus C^* = P^{-1}(R_3' \oplus L_0').$$

Celkově tak můžeme ze zadaných otevřených a šifrových textů:

$$\begin{aligned} T_0 &= L_0 R_0, \qquad & T_0^* &= L_0^* R_0^*, \\ T_3 &= L_3 R_3, \qquad & T_3^* &= L_3^* R_3^* \end{aligned}$$

určit výstupní diference S-boxů po 3. iteraci:

$$C' = P^{-1}(R_3' \oplus L_0') = P^{-1}((R_3 \oplus R_3^*) \oplus (L_0 \oplus L_0^*))$$

a vstupy do S-boxů ve 3. iteraci před aplikací klíče K_3:

$$\begin{aligned} E &= E(R_2) = E(L_3), \\ E^* &= E(R_2^*) = E(L_3^*). \end{aligned}$$

Lze tedy pro všechny S-boxy $S_1, \ldots, S_8$ identifikovat množiny $\mathrm{Test}_j(E_j, E_j^*, C_j')$ obsahující použitou část klíče J_j. Řešení pak pro každý S-box leží v průniku příslušných testových množin získaných z různých zadání dvojic otevřených textů.

Z nalezených částí klíče $J_1, \ldots, J_8$ sestavíme 48-bitový klíč $J = J_1 \ldots J_8$, který byl jako odvozený klíč K_3 použit při šifrování v 3. iteraci. Zpětnou aplikací postupu odvození klíče K_3 určíme 48 bitů šifrového klíče K. Zbývajících 8 šifrových bitů klíče K najdeme probráním $2^8 = 256$ možných kombinací.

Příklad 2.12 *Máme k dispozici 3 dvojice otevřených textů (po vstupní permutaci) tvaru* $T_0 = L_0R_0, T_0^* = L_0^*R_0^*$ *(přičemž* $R_0 = R_0^*$*) a k nim příslušné šifrové texty po 3 iteracích DES tvaru* $T_3 = L_3R_3, T_3^* = L_3^*R_3^*$*:*

	Otevřený text	Šifrový text
1. dvojice	748502CD38451097	03C70306D8A09F10
	3874756438451097	78560A0960E6D4CB
2. dvojice	486911026ACDFF31	45FA285BE5ADC730
	375BD31F6ACDFF31	134F7915AC253457
3. dvojice	357418DA013FEC86	D8A31B2F28BBC5CF
	12549847013FEC86	0F317AC2B23CB944

Z nich odvodíme vstupy do S-boxů před použitím klíče ve 3. iteraci $E = E(L_3), E^* = E(L_3^*)$ *a výstupní diferenci po 3. iteraci* $C' = P^{-1}((R_3 \oplus R_3^*) \oplus (L_0 \oplus L_0^*))$
1. dvojice:

$$E = \texttt{007E0E80680C}$$
$$E^* = \texttt{BF02AC054052}$$
$$C' = \texttt{965D5B67}$$

2. dvojice:

$$E = \texttt{A0BFF41502F6}$$
$$E^* = \texttt{8A6A5EBF28AA}$$
$$C' = \texttt{9C9C1F56}$$

3. dvojice:

$$E = \texttt{EF15068F695F}$$
$$E^* = \texttt{05E9A2BF5604}$$
$$C' = \texttt{D575DB2B}$$

Pro každou dvojici a každý S-box S_j *nyní porovnáme příslušné testové množiny* $Test_j(E_j, E_j^*, C_j')$*, které musí obsahovat použitou část klíče. Pro první S-box* S_1 *například obdržíme*
1. dvojice: $Test_1(000000, 101111, 1001) = \{000000, 000111, 101000, 101111\}$,
2. dvojice: $Test_1(101000, 100010, 1001) = \{101111, 100101, 011100, 010110\}$,
3. dvojice: $Test_1(111011, 000001, 1101) =$
$= \{111111, 101111, 100001, 011011, 010101, 000101\}$.

Pro jednotlivé S-boxy $S_1, \ldots, S_8$ *jsou průniky testových množin získaných*

pomocí daných tří dvojic textů

$$\begin{aligned} J_1 &= 101111 \\ J_2 &= 000101 \\ J_3 &= 010011 \\ J_4 &= 000000 \\ J_5 &= 011000 \\ J_6 &= 000111 \\ J_7 &= 000111 \\ J_8 &= 110001. \end{aligned}$$

Odtud zpětným použitím schématu odvození klíče K_3 dostáváme následující podobu klíče K:

0001101? 0110001? 01?01?0? 1?00100?
0101001? 0000??0? 111?11?? ?100011?

Připomeňme, že bity na 8., 16., ... místě tvoří lichou paritní kontrolu a tedy jsou určeny předchozími 7 bity (např. na 8. místě bude 0). Rozluštěno je nyní 48 bitů z 56 bitů vlastního klíče. Zbylých 8 bitů určíme probráním 256 možností. Nalezený klíč

$$K = \texttt{1A624C89520DEC46}.$$

2.2.6 Nový šifrový standard AES

Reakcí na úspěšné kryptoanalytické útoky proti šifrovému standardu DES bylo v USA přijetí nového šifrového standardu AES (Advanced Encryption Standard) v roce 2002. V soutěži na kryptografický algoritmus AES zvítězil mezi 15 návrhy algoritmus Rijndael vyvinutý v roce 1998, jehož autory jsou J. Daemen a V. Rijmen [6, 7].

Symetrický kryptosystém AES je podobně jako DES smíšená substitučně-transpoziční šifra. Jeho základní verze pracuje s délkou šifrovaného bloku i šifrovacího klíče 128 bitů (algoritmus Rijndael však umožňuje i jinou volbu velikosti bloku a klíče; – v AES se používá standardně blok 128 bitů a klíče délky 128, 192 a 256 bitů).

2.3 Šifrování s veřejným klíčem – metoda RSA

Největším problémem klasických symetrických kryptosystémů je nutnost předávání tajného klíče, který bude sloužit k šifrování zpráv odesílatelem a dešifrování zpráv příjemcem. Přenos utajených zpráv mezi k účastníky tak vyžaduje výměnu $\frac{k(k-1)}{2}$ klíčů nebo vedení utajené komunikace prostřednictvím jednoho důvěryhodného centra. Průlomem v novodobé historii kryptologie byl proto návrh Diffie a Hellmana (z roku 1976)[4] na vytvoření asymetrických kryptosystémů, u nichž je možno šifrovací klíč zveřejnit, neboť z něj nelze dostupnými prostředky odvodit klíč dešifrovací. Adresát utajovaných zpráv tak může poskytnout všem účastníkům, kteří mu chtějí posílat zprávy, jediný klíč, který jim umožní zprávy zašifrovat pro odeslání. Příslušný dešifrovací klíč zůstává pouze adresátovi, takže může být snadněji utajen.

V této části se seznámíme s principy a vlastnostmi nejrozšířenější metody šifrování s veřejným klíčem založené na velkých prvočíslech – metody RSA, kterou popsali v roce 1977 Rivest, Shamir a Adleman. Tato metoda zatím odolala přímým kryptoanalytickým útokům, a proto se využívá stále rozsáhleji (v poslední době také jako základ pro elektronické podepisování zpráv a pro předávání klíčů v symetrických kryptosystémech).

2.3.1 Podstata a použití RSA-kryptosystému

Princip metody RSA spočívá v aplikaci několik staletí starého výsledku modulární aritmetiky, kterým se podrobněji budeme zabývat v části 2.3.2.

V podstatě jde o to, že pro všechna $x < m$ platí

$$(x^i)^j \bmod m = x,$$

jestliže $m = p \cdot q$, kde p a q jsou prvočísla, a přirozená čísla i, j splňují vztah

$$(i \cdot j) \bmod [(p-1)(q-1)] = 1.$$

Operace $y = x^i \bmod m$ je pak užívána pro šifrování (čísla i, m slouží jako tzv. veřejný klíč – šifrovací exponent a modul) a operace $x = y^j \bmod m$ pro dešifrování (j slouží jako tzv. soukromý klíč – dešifrovací exponent).[5]

[4] Podle nově publikovaných informací tento přístup rozpracovala již předtím britská tajná služba.

[5] Jelikož také $(x^j)^i \bmod m = x$, může být dešifrovací exponent j (soukromý klíč účastníka) využit též v rámci elektronického podpisu: Operace $y = x^j \bmod m$ realizuje podpis soukromým klíčem a operace $x = y^i \bmod m$ je jeho ověřením pomocí veřejného klíče.

Popišme si nyní, jak je tato myšlenka použita v RSA-kryptosystému.

Konstrukce RSA-kryptosystému:

1. Najdeme náhodně dvě velká (v současné praxi se používají více jak 100-ciferná) prvočísla p, q a položíme $m = p \cdot q$.

2. Zvolíme přirozená čísla i, j tak, aby

$$(i \cdot j) \bmod [(p-1)(q-1)] = 1.$$

3. Zveřejníme čísla m, i pro šifrování jako veřejný klíč a *utajíme* p, q, j. Číslo j tvoří soukromý klíč.

Jakým způsobem lze nalézt čísla p, q, i a j popíšeme podrobněji v dalších odstavcích. Podstatné je, že pro nalezení soukromého klíče j z čísel m, i neexistuje rychlý algoritmus, neboť problém rozkladu m na prvočísla p, q je výpočetně velmi složitý. Vzniklý RSA-kryptosystém má tak pro dostatečně velká prvočísla p, q vlastnost asymetrického kryptosystému: nalezení dešifrovacího klíče ze šifrovacího je dostupnými prostředky nezvládnutelné.

Použití RSA-kryptosystému:

Šifrování: Zpráva M (nebo její část – dlouhé zprávy musíme samozřejmě rozložit na bloky) je nejdříve zakódována jako číslo $x < m$ (např. bitový řetězec zprávy je chápán jako zápis čísla ve dvojkové soustavě). Poté je vypočítáno šifrové číslo

$$y = x^i \bmod m.$$

Pro výpočet i-té mocniny čísla x existuje v modulární aritmetice rychlá procedura postupného výpočtu čtverců čísel modulo m. (Tuto proceduru popíšeme níže.) Vlastní šifrou zprávy M je šifrové číslo y (převedené třeba opět na binární zápis).

Dešifrování: Ze šifrového čísla $y < m$ určíme kód zprávy – číslo $x < m$:

$$x = y^j \bmod m,$$

přičemž opět použijeme proceduru postupného výpočtu čtverců čísel modulo m. Poté x převedeme do binárního zápisu na bitový řetězec zprávy.

Příklad 2.13 *Konstrukci a použití RSA-kryptosystému předvedeme na miniaturním ilustrativním příkladu:*

1. *Zvolíme prvočísla* $p = 5, q = 17$ *a určíme modul* $m = 85$.

2. *Zvolíme exponenty* i, j*:*
   $(p-1)(q-1) = 4 \cdot 16 = 64$, *takže můžeme zvolit* $i = 5$ *a* $j = 13$, *neboť* $(5 \cdot 13) = 1 \bmod 64$.

3. *Zveřejníme* $m = 85, i = 5$ *pro šifrování, utajíme soukromý klíč* $j = 13$.

Šifrování: *Můžeme šifrovat 6-bitové zprávy (binární čísla* < 64*).*
Např. zprávě 101101 *chápané jako zápis čísla ve dvojkové soustavě odpovídá* $x = 45$*. Šifrové číslo určíme jako*

$$y = 45^5 \bmod 85 = 10,$$

neboť

$$45^5 = 184528125 = 2170919 \cdot 85 + 10$$

Po převodu na binární zápis dostáváme šifru 001010.

Dešifrování: *Šifra* 001010 *představuje v decimálním zápise číslo* $y = 10$.
K dešifrování použijeme soukromý klíč $j = 13$:

neboť

$$x = 10^{13} \bmod 85 = 45,$$

$$10^{13} = 10000000000000 = 117647058823 \cdot 85 + 45$$

Dešifrovaná zpráva je binární zápis čísla 45*:* 101101.

V následujícím příkladu se seznámíme s rychlou procedurou výpočtu velkých mocnin modulo m pomocí metody postupného výpočtu čtverců čísel modulo m.

Příklad 2.14 *V RSA-kryptosystému s modulem* $m = 11413$ *a veřejným (šifrovacím) exponentem* $i = 3533$ *provedeme zašifrování zprávy kódované číslem* $x = 9726$*, tj. výpočet*

$$y = 9726^{3533} \bmod 11413.$$

Jelikož exponent 3533 *je v binárním vyjádření roven*

$$3533 = 2^{11} + 2^{10} + 2^8 + 2^7 + 2^6 + 2^3 + 2^2 + 2^0,$$

můžeme mocninu vyjádřit jako

$$x^{3533} = x^{2^{11}} \cdot x^{2^{10}} \cdot x^{2^8} \cdot x^{2^7} \cdot x^{2^6} \cdot x^{2^3} \cdot x^{2^2} \cdot x^{2^0},$$

Vypočteme proto postupně pro $k = 0, \ldots, 11$ *čísla* $y_k = x^{2^k} \bmod m$ *s využitím rovnosti* $y_k = y_{k-1}^2 \bmod m$*:*

$$
\begin{aligned}
y_0 &= 9726^1 \bmod 11413 = 9726,\\
y_1 &= 9726^2 \bmod 11413 = 4132,\\
y_2 &= 9726^4 \bmod 11413 = 4132^2 \bmod 11413 = 10989,\\
y_3 &= 9726^8 \bmod 11413 = 10989^2 \bmod 11413 = 8581,\\
y_4 &= 9726^{16} \bmod 11413 = 8581^2 \bmod 11413 = 8298,\\
y_5 &= 9726^{32} \bmod 11413 = 8298^2 \bmod 11413 = 2175,\\
y_6 &= 9726^{64} \bmod 11413 = 2175^2 \bmod 11413 = 5643,\\
y_7 &= 9726^{128} \bmod 11413 = 5643^2 \bmod 11413 = 1179,\\
y_8 &= 9726^{256} \bmod 11413 = 1179^2 \bmod 11413 = 9068,\\
y_9 &= 9726^{512} \bmod 11413 = 9068^2 \bmod 11413 = 9372,\\
y_{10} &= 9726^{1024} \bmod 11413 = 9372^2 \bmod 11413 = 11349,\\
y_{11} &= 9726^{2048} \bmod 11413 = 11349^2 \bmod 11413 = 4096.
\end{aligned}
$$

Nyní tedy dostáváme

$$y = 9726^{3533} \bmod 11413 = (y_{11} \cdot y_{10} \cdot y_8 \cdot y_7 \cdot y_6 \cdot y_3 \cdot y_2 \cdot y_0) \bmod 11413.$$

Zbývající výpočet můžeme opět provést postupně:

$$
\begin{aligned}
y' &= (y_2 \cdot y_0) \bmod 11413 = (9726 \cdot 10989) \bmod 11413 = 7682,\\
y'' &= (y_3 \cdot y') \bmod 11413 = (7682 \cdot 8581) \bmod 11413 = 9167,\\
y''' &= (y_6 \cdot y'') \bmod 11413 = (9167 \cdot 5643) \bmod 11413 = 5665,\\
y'''' &= (y_7 \cdot y''') \bmod 11413 = (5665 \cdot 1179) \bmod 11413 = 2430,\\
y''''' &= (y_8 \cdot y'''') \bmod 11413 = (2430 \cdot 9068) \bmod 11413 = 8150,\\
y'''''' &= (y_{10} \cdot y''''') \bmod 11413 = (8150 \cdot 11349) \bmod 11413 = 3398,\\
y &= (y_{11} \cdot y'''''') \bmod 11413 = (3398 \cdot 4096) \bmod 11413 = 5761.
\end{aligned}
$$

Stejnou proceduru může využít příjemce pro dešifrování šifry $y = 5761$*. Jako soukromý exponent je v tomto RSA-kryptosystému* $j = 6597$ *(viz příklad 2.16), takže jde o výpočet*

$$x = y^j \bmod m = 5761^{6597} \bmod 11413.$$

Provedení výpočtu přenecháme čtenáři jako cvičení.

2.3.2 Matematické principy – Eulerova věta

Metoda šifrování s veřejným klíčem založená na velkých prvočíslech vychází z řady tvrzení teorie čísel, která tak našla nečekané a rozsáhlé využití v praxi.

Základem pro jejich formulaci je zavedení *Eulerovy funkce* $\varphi(m)$, která pro přirozené číslo m udává počet čísel menších než m nesoudělných s m. Pro hodnoty Eulerovy funkce $\varphi(m)$ platí následující tvrzení, která můžeme využít pro její výpočet:

VĚTA 2.1 *Nechť p a q jsou různá prvočísla. Pak pro Eulerovu funkci φ platí*

$$\varphi(p) = p - 1$$
$$\varphi(p^i) = (p-1)p^{i-1},$$
$$\varphi(p \cdot q) = (p-1)(q-1).$$

Jsou-li $p_1, \ldots, p_k$ různá prvočísla, pak

$$\varphi(p_1^{i_1} \ldots p_k^{i_k}) = (p_1 - 1)p_1^{i_1 - 1} \ldots (p_k - 1)p_k^{i_k - 1}.$$

Jsou-li m, n nesoudělná přirozená čísla, pak

$$\varphi(m \cdot n) = \varphi(m) \cdot \varphi(n).$$

Zásadní význam pro metodu RSA má *Eulerova věta*:

VĚTA 2.2 *Jestliže a, m jsou* nesoudělná *přirozená čísla, pak*

$$a^{\varphi(m)} \bmod m = 1.$$

Jejím speciálním případem je věta dokázaná dříve Fermatem a nazývaná *malá Fermatova věta*:

VĚTA 2.3 *Jestliže p je prvočíslo, potom pro každé přirozené číslo b nesoudělné s p je*

$$b^{p-1} \bmod p = 1.$$

Vlastní princip metody RSA je založen na následujícím již dříve zmíněném tvrzení, které vyplývá z Eulerovy věty.

VĚTA 2.4 *Nechť i, j, m jsou přirozená čísla taková, že i je nesoudělné s $\varphi(m)$ a*

$$(i \cdot j) \bmod \varphi(m) = 1.$$

Potom pro libovolné přirozené číslo x nesoudělné s m platí

$$(x^i)^j \bmod m = x \bmod m.$$

Dosadíme-li do této věty $m = p \cdot q$ pro různá prvočísla p, q a ověříme i případy $x = p, x = q$, dostaneme výsledek používaný pro šifrování a dešifrování v RSA.

2.3.3 Generování velkých prvočísel

První krok při konstrukci RSA-kryptosystému spočívá v nalezení dvou velkých prvočísel p, q (v současné době jsou požadována zhruba 100-ciferná prvočísla). Kolik takových prvočísel je a jak rychle najít potřebná dvě?

V teorii čísel byla již před staletími intenzívně zkoumána funkce

$$\pi(n) = \text{počet prvočísel menších než } n.$$

Základní odhad této funkce podal Gauss:

$$\pi(n) \sim \frac{n}{\ln n}.$$

Už z prvního Gaussova odhadu vyplývá, že relativní frekvence výskytu prvočísel v okolí čísla n je přibližně $\frac{1}{\ln n}$ a průměrná vzdálenost $A(n)$ mezi dvěma prvočísly v okolí čísla n je tedy $A(n) \sim \ln n$.

Příklad 2.15 *Průměrná vzdálenost $A(n)$ mezi dvěma prvočísly v okolí některých čísel:*

$$A(20) = \ln 20 \sim 3,$$
$$A(150) = \ln 150 \sim 5,$$
$$A(10^{50}) = \ln 10^{50} \sim 115,$$
$$A(10^{100}) = \ln 10^{100} \sim 230.$$

100-ciferná prvočísla tedy za sebou následují v průměru po 230 číslech. Je jich $\pi(10^{100}) - \pi(10^{99})$, což je přibližně $3.9 \cdot 10^{97}$. Odtud vyplývá, že lze efektivně hledat i velká prvočísla náhodně. U 100-ciferných čísel bude v průměru stačit projít 115 za sebou jdoucích čísel počínaje náhodně zvoleným. U každého z nich se rychlými procedurami testuje, zda není dělitelné malými prvočísly (sudá čísla a čísla končící cifrou 5 se rovnou přeskakují, dělitelnost 3 se ověří snadno pomocí ciferného součtu, apod.) a zda je pro různé případy splněna rovnost z malé Fermatovy věty 2.3 (není-li pro některý případ splněna, nejedná se o prvočíslo). Princip tzv. Rabinova algoritmu můžeme shrnout takto:

1. Generuje se náhodně liché 100-ciferné číslo n.

2. Provádí se k pokusů ověřujících, zda je pro vhodně vybraná $a < n$ splněna rovnost z malé Fermatovy věty, tj. zda $a^{n-1} \bmod n = 1$.
   Pokud pro některé a rovnost neplatí, je n jistě číslo složené. V tom případě polož $n = n + 2$ a opakuj krok 2.

3. Číslo n, které prošlo dostatečným počtem k pokusů, je s velkou pravděpodobností prvočíslo (což může být dále testováno složitějšími testy prvočíselnosti).

Jde o pravděpodobnostní algoritmus: o výsledném čísle nevíme s absolutní jistotou, ale pouze s vysokou pravděpodobností, že je prvočíslem. Důležité však je, že realizované verze algoritmu pracují rychle (s tzv. logaritmickou složitostí – počet kroků je omezen násobkem logaritmu čísla n) a je prokázáno, že pravděpodobnost špatného výsledku (výsledné číslo n je přece jen složené) exponenciálně rychle klesá k nule v závislosti na počtu a druhu prováděných testů prvočíselnosti. V základní verzi metody je tato pravděpodobnost menší než $\frac{1}{2^k}$. Po úspěšném provedení deseti ověřovacích pokusů je tedy více jak 99,9% pravděpodobnost, že n je prvočíslo.

2.3.4 Určení veřejného a soukromého klíče

Druhým krokem při konstrukci RSA-kryptosystému je určení šifrovacího exponentu i a dešifrovacího exponentu j tak, aby byla splněna podmínka základní věty metody RSA

$$(i \cdot j) \bmod \varphi(m) = 1.$$

VĚTA 2.5 *Jestliže i není soudělné s $\varphi(m)$, pak existuje právě jedno $j < \varphi(m)$ takové, že*

$$(i \cdot j) \bmod \varphi(m) = 1.$$

Číslo j je určeno formulí

$$j = i^{\varphi(\varphi(m))-1} \bmod \varphi(m).$$

Z věty o určení exponentů v RSA $\varphi(m) = (p-1)(q-1)$, je možno určit exponent j z exponentu i podle formule

$$j = i^{\varphi((p-1)(q-1))-1} \bmod (p-1)(q-1).$$

Přitom $\varphi((p-1)(q-1))$ vypočteme v našich příkladech pomocí rozkladu $(p-1)(q-1)$ na prvočinitele. Veřejný šifrovací exponent i přitom volíme tak, aby $3 \leq i < (p-1)(q-1)$, aby nebyl soudělný s $(p-1), (q-1)$ a aby $i^k \bmod (p-1)(q-1) \neq 1$ pro "malé" k.

Příklad 2.16 *V příkladu 2.14, kde* $m = 101 \cdot 113$*, dostáváme podle věty 2.1*

$$\varphi(\varphi(m)) = \varphi((101-1)\cdot(113-1)) = \varphi(11200)$$
$$= \varphi(2^6 \cdot 5^2 \cdot 7^1) = 2^5 \cdot 4 \cdot 5 \cdot 6 = 3840.$$

Můžeme tedy určit exponent j *z exponentu* i *pomocí formule z věty 2.5:*

$$j = i^{3839} \bmod 11200.$$

Úloha 2.1 *Vypočtěte metodou postupného výpočtu čtverců exponent* j *pro* $i = 3533$*, tj.*

$$j = 3533^{3839} \bmod 11200.$$

2.3.5 Faktorizace a kryptoanalýza RSA

Každý asymetrický kryptosystém je postaven na předpokladu, že z veřejného šifrovacího klíče nelze dostupnými prostředky sestrojit tajný dešifrovací klíč. V RSA-kryptosystému je veřejný klíč tvořen modulem m a šifrovacím exponentem i. Pokud by se útočníkovi podařilo nalézt rozklad modulu m na dvě prvočísla p, q, pak již dešifrovací exponent j najde podle věty 2.5:

$$j = i^{\varphi((p-1)\cdot(q-1))-1} \bmod (p-1)(q-1).$$

Příklad 2.17 *Najde-li útočník rozklad modulu* $m = 11413$ *z příkladu 2.14, pak určí dešifrovací exponent* j *z veřejného exponentu* $i = 3533$ *postupem z příkladu 2.16:*

$$j = 3533^{3839} \bmod 11200 = 6597.$$

Odolnost RSA-kryptosystému je tedy principiálně dána neřešitelností problému rozkladu modulu m dostupnými prostředky. Obecně jde o problém rozkladu čísla na prvočinitele – tzv. problém faktorizace.

Podle jedné ze základních vět aritmetiky existuje ke každému přirozenému číslu m jeho jednoznačný rozklad

$$m = p_1^{k_1} \dots p_r^{k_r},$$

kde $p_1 < \dots < p_r$ jsou prvočísla a $k_1, \dots, k_r$ jsou přirozená čísla.

Problémem je však efektivní nalezení tohoto rozkladu. Výpočetní složitost problému faktorizace můžeme ilustrovat na novodobé historii úspěšných rozkladů (viz [14]): V roce 1970 byla rozkládána 20-ciferná čísla, v roce 1980 umožnil rozvoj počítačů a vývoj metody řetězových zlomků rozklad 50-ciferných čísel, v dalším desetiletí metoda kvadratického síta vedla k rozkladu 100-ciferných čísel. V polovině 90. let využití distribuovaného zpracování prostřednictvím Internetu dovolilo rozklad konkrétního 129-ciferného čísla. Nejnovější metoda – tzv. síto číselného tělesa – umožňovala kolem roku 1996 rozklad 130-ciferných čísel. Stav v roce 2003 je zřejmý z toho, že nejmenší číslo zařazené do soutěže RSA-Challenge v roce 2003 mělo 174 cifer (viz příklad 2.18).

Příklad 2.18 *Na ukázku uvádíme rozklad 174-ciferného čísla, na nějž byla v roce 2003 vypsána odměna 10 tis. $. Koncem roku 2003 se rozklad podařilo najít (týmu ve složení Franke, Kleinjung, Montgomery, te Riele, Bahr, Leclair, Leyland, Wackerbarth):*

$$
\begin{aligned}
&188198812920607963838697239461650439807163563379417382700763356422988859715234665485319060606504743045317388011303396716199692321205734031879550656996221305168759307650257059\\
&=\\
&398075086424064937397125500550386491199064362342526708406385189579946388957261768583317\\
&\times\\
&472772146107435302536223071973048224632914695302097116459852171130520711256363590397527.
\end{aligned}
$$

2.4 Další metody šifrování s veřejným klíčem

V této části stručně uvedeme další metody šifrování, které představují podobně jako metoda RSA realizaci principu šifrování s veřejným klíčem. Bezpečnost těchto metod je založena na obtížnosti řešení jiných úloh, než byla faktorizace u metody RSA.

2.4.1 Diffie-Hellmanova metoda výměny klíčů

Diffie-Hellmanova metoda výměny klíčů byla autory publikována již v roce 1976 [9]. Jde o to, že účastníci komunikace si prostřednictvím veřejného kanálu vymění informace, které jim umožní sestavit společný tajný klíč pro šifrování vzájemné výměny zpráv.

Účastníci se nejdříve dohodnou na velkém prvočísle q a primitivním prvku $g < q$, tj. prvku, jehož mocniny $g^k \bmod q$ pro různá k nabývají všech hodnot $1, \ldots, q-1$. Čísla q, g mohou být veřejně známa, a proto se na nich účastníci mohou dohodnout prostřednictvím veřejného kanálu.

V dalším kroku si každý účastník i volí tajně nenulový prvek $s_i < q$ jako soukromý klíč. Jako veřejný klíč oznámí ostatním účastníkům

$p_i = g^{s_i} \bmod q$.

Pokud chce dvojice účastníků i, j bezpečně komunikovat, může každý z dvojice vypočítat z jemu známých čísel hodnotu[6]

$$K_i = (p_j)^{s_i} \bmod q = g^{s_j s_i} \bmod q,$$

$$K_j = (p_i)^{s_j} \bmod q = g^{s_i s_j} \bmod q.$$

Tyto hodnoty jsou stejné, $K_i = K_j = g^{s_i s_j} \bmod q$ tak tvoří společný tajný klíč, který je pak účastníky používán v některém symetrickém kryptosystému pro zabezpečení jejich vzájemné komunikace.

Příklad 2.19 *Pro ilustraci použití Diffie-Hellmanovy metody výměny klíčů mezi dvěma účastníky zvolíme malá čísla* $q = 17$, $g = 3$. *Řekněme, že účastník* i *tajně zvolí číslo* $s_i = 7$ *a spočte svůj veřejný klíč*

$p_i = g^{s_i} \bmod q = 3^7 \bmod 17 = 11$,

podobně účastník j *tajně zvolí číslo* $s_j = 4$ *a spočte svůj veřejný klíč*

$p_j = g^{s_j} \bmod q = 3^4 \bmod 17 = 13$.

Účastníci si tyto veřejné klíče vymění a s využitím své tajné části vypočítají společný klíč:

Účastník i *:* $K_i = (p_j)^{s_i} \bmod q = 13^7 \bmod 17 = 4$.

[6]Připomeňme, že pro výpočet mocnin existuje rychlá procedura postupného výpočtu čtverců čísel modulo q (viz příklad 2.14).

Účastník j : $K_j = (p_i)^{s_j} \bmod q = 11^4 \bmod 17 = 4$.
Vidíme, že oba účastníci určí klíč, který je roven
$K = g^{s_i s_j} \bmod q = 3^{7 \cdot 4} \bmod 17 = 4$.

Bezpečnost algoritmu spočívá v obtížné řešitelnosti problému diskrétního logaritmu – neexistuje totiž efektivní algoritmus, jak k danému číslu X najít x takové, že $X = g^x \bmod q$. Je-li tedy q *dost velké*, nemůže nikdo vypočítat s_i ze znalosti g a p_i.

2.4.2 Metoda založená na problému batohu

Problém batohu (knapsack problem) je známá optimalizační úloha, jejíž typické aplikace se týkají např. plánování efektivního využití kontejnerové přepravy či nejcennějšího nákladu lodi. Setkal se s ním ovšem každý, kdo stál před úkolem sbalit si batoh na delší výlet. Obecná formulace zní: Je dáno přirozené číslo S – celkový objem batohu a přirozená čísla

$$\begin{aligned} a_1, \ldots, a_k &- \text{ objemy jednotlivých předmětů,} \\ c_1, \ldots, c_k &- \text{ ceny těchto předmětů.} \end{aligned}$$

Optimalizační úlohou je nalézt hodnoty binárních proměnných $x_1, \ldots, x_k$, které maximalizují celkovou cenu nákladu

$$x_1 c_1 + \ldots + x_k c_k \longrightarrow MAX$$

za podmínky nepřekročení objemu

$$x_1 a_1 + \ldots + x_k a_k \leq S.$$

Kombinatorickým jádrem problému batohu je otázka, zda lze z předmětů s danými objemy vybrat podmnožinu právě naplňující batoh s daným celkovým objemem, tj. nalézt bitový řetězec $x_1 \ldots x_k$ takový, že

$$x_1 a_1 + \ldots + x_k a_k = S.$$

Tato úloha patří spolu s velkým množstvím jiných kombinatorických úloh do třídy tzv. NP-úplných problémů, pro něž není znám efektivní algoritmus řešení (pracující s polynomiální výpočetní složitostí v závislosti na velikosti vstupu – zde počtu předmětů). Přitom by sestavení efektivního algoritmu pro řešení jedné z NP-úplných úloh znamenalo, že bude takový algoritmus sestrojitelný i pro všechny ostatní úlohy této třídy.

Obecný problém batohu je proto považován za efektivně neřešitelný, což je základem pro metodu šifrování s veřejným klíčem, kterou navrhli v

roce 1978 Merkle a Hellman. Ta současně využívá i skutečnosti, že existují speciální třídy tohoto problému, které jsou řešitelné velice rychle – tzv. snadné problémy batohu. Například problém, v němž dané objemy předmětů $a'_1, \ldots, a'_k$ tvoří superklesající posloupnost, tj. mají vlastnost

$$a'_1 > \sum_{l=2}^{k} a'_l, \quad a'_2 > \sum_{l=3}^{k} a'_l, \ \ldots, \ a'_{k-2} > a'_{k-1} + a'_k, \quad a'_{k-1} > a'_k$$

jsou řešitelné velice snadno, dokonce v lineárním čase: do batohu postupně přidáváme vždy první předmět ve zbývající řadě, který se ještě vejde.

Příklad 2.20 *Jednoduchým příkladem superklesající posloupnosti délky k je $a'_l = 2^{k-l}$, $l = 1, \ldots, k$. Řešením úlohy batohu pro objem $S \leq 2^k - 1$ je pak binární zápis čísla S.*
Např. pro $k = 6$ je $(a'_1, \ldots, a'_k) = (32, 16, 8, 4, 2, 1)$ superklesající posloupnost a třeba pro $S = 52$ má příslušná úloha batohu

$$52 = x_1 \cdot 32 + x_2 \cdot 16 + x_3 \cdot 8 + x_4 \cdot 4 + x_5 \cdot 2 + x_6 \cdot 1$$

řešení $x_1x_2x_3x_4x_5x_6 = 110100$.

Myšlenka postupu šifrování na základě problému batohu tedy tkví v tom, že zašifrovanou zprávu tvoří zadání obtížné formy tohoto problému. Dešifrovací tajné klíče pak umožní převedení problému na problém snadný, jehož řešení je shodné s řešením problému obtížného. Šifrovací metodu popíšeme podle [1].

Šifrovaný binární řetězec $x_1 \ldots x_k$ šifrujeme jako sumu

$$S = x_1a_1 + \ldots + x_ka_k$$

pomocí veřejného klíče tvořeného čísly $a_1, \ldots, a_k$. Při dešifrování nalezneme řešení úlohy batohu $S, a_1, \ldots, a_k$, tím, že úlohu níže popsaným způsobem převádíme na určitý snadný problém batohu.

Konstrukce kryptosystému na základě problému batohu

1. Zvolíme snadný problém batohu $a'_1, \ldots, a'_k$.

2. Zvolíme číslo $m > a'_1 + \ldots + a'_k$, dále číslo v nesoudělné s m a k němu najdeme inverzní w, které bude sloužit k dešifrování:

$$v \cdot w = 1 \bmod m.$$

3. Sestrojíme transformovaný (obtížný) problém batohu

$$\begin{aligned} a_1 &= v \cdot a'_1 \bmod m \\ &\vdots \\ a_k &= v \cdot a'_k \bmod m \end{aligned}$$

Pomocné číslo v utajíme, stejně jako soukromý dešifrovací klíč w, m, a zveřejníme $(a_1, \ldots, a_k)$.

Šifrování: Zpráva: $x_1 \ldots x_k$ je šifrována sumou $S = x_1 a_1 + \ldots + x_k a_k$.

Dešifrování: Určíme snadnou formu problému:

$$\begin{aligned} S' &= wS \bmod m, \\ a'_1 &= wa_1 \bmod m, \\ &\vdots \\ a'_k &= wa_1 \bmod m, \end{aligned}$$

jejíž řešení dostaneme takto:

$$\begin{aligned} &x_1 = 1, \text{ když } S' \geq a'_1; \\ &x_2 = 1, \text{ když } S' - x_1 a'_1 \geq a'_2; \\ &x_3 = 1, \text{ když } S' - x_1 a'_1 - x_2 a'_2 \geq a'_3; \\ &\text{atd.} \end{aligned}$$

Dešifrování dává správný výsledek – nalezené řešení je též řešením transformovaného (obtížného) problému batohu daného objemy $(a_1, \ldots, a_k)$, S, neboť platí

$$\begin{aligned} x_1 a_1 + \ldots + x_k a_k &= x_1(va'_1 \bmod m) + \ldots + x_k(va'_k \bmod m) \\ &= v \cdot (x_1 a'_1 + \ldots + x_k a'_k) \bmod m = v \cdot S' \bmod m = v \cdot w \cdot S \bmod m = S. \end{aligned}$$

Příklad 2.21 *Ke konstrukci ilustrativního kryptosystému využijeme snadný problém batohu z předchozího příkladu:* $(32, 16, 8, 4, 2, 1)$.

Zvolíme $m = 89 > 32 + 16 + 8 + 4 + 2 + 1$ *a* $v = 30$. *Určíme inverzní prvek* $w = 3$, *pro nějž* $30 \cdot 3 = 1 \bmod 89$.

Nyní vypočteme transformovaný problém batohu:

$$\begin{aligned} a_1 &= 30 \cdot 32 \bmod 89 = 70, \\ a_2 &= 30 \cdot 16 \bmod 89 = 45, \\ a_3 &= 30 \cdot 8 \bmod 89 = 62, \\ a_4 &= 30 \cdot 4 \bmod 89 = 31, \\ a_5 &= 30 \cdot 2 \bmod 89 = 60, \\ a_6 &= 30 \cdot 1 \bmod 89 = 30. \end{aligned}$$

Zveřejníme tedy čísla $(70, 45, 62, 31, 60, 30)$. *Např. zpráva* $M = 101010$ *bude v tomto kryptosystému zašifrována sumou:* $S = 70 + 62 + 60 = 192$.

Její dešifrování spočívá v transformaci obtížného problému batohu pomocí soukromého klíče $(w, m) = (3, 89)$*:*

$$\begin{aligned} S' &= w \cdot S \bmod m = 3 \cdot 192 \bmod 89 = 42, \\ a'_1 &= 3 \cdot 70 \bmod 89 = 32, \\ &\vdots \\ a'_6 &= 3 \cdot 30 \bmod 89 = 1 \end{aligned}$$

na snadný a jeho vyřešení

$$42 = x_1 \cdot 32 + x_2 \cdot 16 + x_3 \cdot 8 + x_4 \cdot 4 + x_5 \cdot 2 + x_6 \cdot 1.$$

Výsledkem je $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 1, x_6 = 0$, *což dává původní zprávu* $M = 101010$.

2.5 Cvičení ke kapitole 2

1. Pokuste se pomocí Kasiského testu a metody koincidencí rozluštit následující text (vzniklý Vigenèrovou šifrou z anglického textu s vynecháním mezer):

   ```
   OOBQBPQAIUNEUSRTEKASRUMNARRMNRROPYODEEADERUNRQLJUGCZCCUNRTEU
   ARJPTMPAWUTNDOBGCCEMSOHKARCMNBYUATMMDERDUQFWMDTFKILROPYARUOL
   FHYZSNUEQMNBFHGEILFEJXIEQNAQEVQRREGPQARUNDXUCZCCGPMZTFQPMXIA
   UEQAREAVCDNKQNREYCFIFTAQZETQRFMDYOHPANGOLCD
   ```

   (Poznamenejme, že ke konci textu se vyskytuje chyba vzniklá při ručním šifrování či přepisování šifrového textu.)

2. Změňte v klíči z příkladu 2.8 některý bit a sledujte na příkladu 2.9 v průběhu šifrování standardem DES, jak se změna projevuje.

3. Zvolíme-li prvočísla $p = 5$, $q = 13$, pak můžeme pro aplikaci metody RSA stanovit např. exponenty $i = 5$ a $j = 29$, neboť

   $$5 \cdot 29 \bmod 48 = 1.$$

   Ukažte, že řetězec 001010 (tedy číslo 10) je v tomto případě zašifrováno řetězcem 011110 (číslem 30). S použitím rychlé procedury výpočtu velkých mocnin ukažte, že je toto číslo správně dešifrováno, tedy, že

   $$30^{29} \bmod 65 = 10.$$

4. Proveďte kryptoanalýzu RSA-kryptosystému s veřejným klíčem $m = 209$, $i = 7$ a dešifrujte zprávu $y = 29$

5. Vyzkoušejte Diffie-Hellmanovu výměnu klíčů pro $q = 11$, $g = 2$, $s_i = 4$, $s_j = 3$.

6. Dešifrujte v kryptosystému z Příkladu 2.21 zprávu zašifrovanou sumou $S = 136$.

Literatura

[1] Adámek, J.: *Kódování.* SNTL, Praha 1989.

[2] Beker H.J., Piper F.C.: *Communications security. A survey of cryptography.* IEEE Proc. 129 (1982), Pt. A, No. 6, str. 357–376.

[3] Buchmann J.A.: *Introduction to Cryptography.* Springer Verlag, New York 2002.

[4] Černý, J.: *Entropia a informácia v kybernetike.* Bratislava, Alfa 1981.

[5] Churchhouse R.: *Codes and Ciphers. University Press* Cambridge 2002.

[6] Daemen J., Rijmen V.: *AES Proposal: Rijndael.* 45 str. (http://CSRC.NIST.GOV/CRYPTOTOOLKIT/AES/RIJNDAEL).

[7] Daemen J., Rijmen V.: *The Design of Rijndael. AES – The Advanced Encryption Standard.* Springer Verlag 2002.

[8] Delfs H., Knebl H.: *Introduction to Cryptography. Principles and Applications.* Springer Verlag 2002. xiv + 310 str.

[9] Diffie W., Hellman M.E.: *New directions in cryptography.* IEEE Trans. Inform. Theory IT-22 (1976), str.644–654.

[10] Dobda L.: *Ochrana dat v informačních systémech.* Grada, Praha 1998.

[11] Gallager R.G.: *Information Theory and Realible Communications.* New York, 1968.

[12] Paseka J.: *Kryptografie.* Masarykova universita Brno 11.2.1998, 220 str. (http://www.mu.cz/).

[13] Pelikán, J.: *Teorie informace.* Praha, VŠE 1970.

[14] Pomerance, C.: Vyprávění o dvou sítech. Pokroky matematiky, fyziky a astronomie 43 (1998), č.1, str. 9-29

[15] Schneier B.: *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley 1996.

[16] Seberry J., Pieprzyk J.: *Cryptography. An Introduction to Computer Security.* Prentice Hall 1989.

[17] Singh S.: *Kniha kódů a šifer.* Dokořán, Praha 2003.

[18] Stinson D.R.: *Cryptography: Theory and Practice.* CRC Press 1995.

[19] Trappe W., Washington L.C.: *Introduction to Cryptography with Coding Theory.* Prentice Hall 2002.

[20] Vajda, I.: *Teória informácie a štatistického rozhodovania.* Bratislava, Alfa 1982.