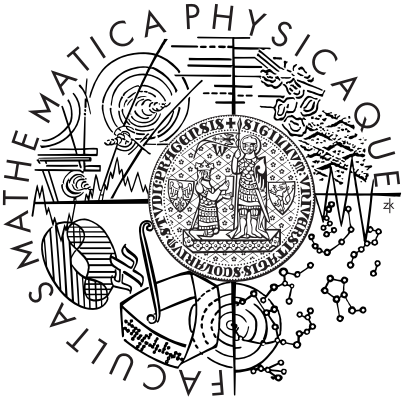
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Adam Christov

Statistické testy hašovacích (=žvýkacích) funkcí

Katedra algebry

Vedoucí bakalářské práce: Doc. RNDr. Jiří Tůma, DrSc.

Studijní program: Matematika
Studijní obor: Obecná matematika

2006

Chtěl bych poděkovat svému vedoucímu Doc. RNDr. Jiřímu Tůmovi, DrSc. za zadání práce a pomoc při jejím tematickém uspořádaní a také svým spolužákům, především Jakubu Pečánkovi, za cenné rady ke statistické části práce a pomoc při hledání stylistických a gramatických chyb.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 10. srpna 2006 Adam Christov

Obsah

Název práce: Statistické testy hašovacích (=žvýkacích) funkcí
Autor: Adam Christov
Katedra: Katedra algebry
Vedoucí bakalářské práce: Doc. RNDr. Jiří Tůma, DrSc.
e-mail vedoucího: tuma@karlin.mff.cuni.cz

Abstrakt: V předložené práci popisujeme obecnou konstrukci hašovacích funkcí a podrobněji hašovací funkce MD5. Ukazujeme metodu Wangové a kol. na hledání kolizí hašovací funkce MD5 a popisujeme algoritmy Vlastimila Klímy a Marca Stevense, které jsou na této metodě založené. Dále pak provádíme jednoduché statistické testy, kterými se snažíme ověřit náhodnost výstupu MD5 a naplnění postačujících podmínek, které se využívají v obou zmíněných algoritmech na hledání kolizí.
Klíčová slova: MD5, hašovací funkce, kolize, statistické testy

Title: Statistical testing of hash functions
Author: Adam Christov
Department: Department of Algebra
Supervisor: Doc. RNDr. Jiří Tůma, DrSc.
Supervisor's e-mail address: tuma@karlin.mff.cuni.cz

Abstract: In this paper we describe the general workings of hash functions and deal with MD5 hash function in more detail. We present Wang et al. method for finding collisions in MD5 hash function and we describe Vlastimil Klíma's and Marc Stevens's algorithms that are based on this method. We also perform simple statistical tests to verify randomness of MD5 output and fulfilment sufficient conditions which are used in both mentioned algorithms for finding collisions.

Keywords: MD5, hash functions, collision, statistical testing

Úvod

V této práci se budeme zabývat moderními hašovacími funkcemi, konkrétně pak stále hojně používanou hašovací funkcí MD5. Nejprve stručně shrneme základní požadované vlastnosti hašovacích funkcí a jejich využití v praxi. Dále si uvedeme, jak jsou hašovací funkce obecně konstruovány, podrobně si popíšeme konstrukci hašovací funkce MD5 a provedeme její statistické testy. Ukážeme si principy útoků na MD5 a statisticky otestujeme některé předpoklady úspěšnosti těchto útoků.

Hašovací funkce je funkce, která libovolně dlouhé posloupnosti bitů (*zprávě*) přiřadí řetězec bitů o pevně dané délce (*haš* nebo také *digitální otisk*). Přitom by mělo být výpočetně jednoduché ji spočítat a naopak výpočetně nemožné na základě haše určit jakýkoliv její vzor (*jednosměrnost*). Další požadovanou vlastností je nemožnost nalézt jakékoliv dvě zprávy se stejnou haší (*bezkoliznost*). Pro zamezení hledání kolizí a vzorů hrubou silou je potřeba, aby byl výstup hašovací funkce jistým způsobem náhodný. Tím se myslí, aby jakékoliv posloupnosti různých zpráv odpovídala náhodná posloupnost prvků (haší) z množiny všech možných obrazů.

Hašovací funkce mají široké využití. Používají se například při ukládání a ověřování přihlašovacích hesel. Místo toho, aby se ukládalo samotné heslo, uloží se jeho haš. Tím se znemožní odhalení hesla v případě přístupu k úložišti (ať už administrátora nebo nepověřené osoby), jelikož z haše je nemožné díky jednosměrnosti zpětně zjistit původní heslo. Při kontrole se pak jednoduše spočítá haš zadaného hesla, která se porovná s dříve uloženou hodnotou. Hašovací funkce jsou také důležitou součástí kryptografických systémů pro digitální podpisy. Vzhledem k bezkoliznosti stačí místo zprávy podepsat její haš. Znatelně se tak proces podepisování urychlí, protože místo několika kilobajtové zprávy se podepisuje její haš o velikosti několika stovek bitů. Dále se hašovací funkce využívají například při kontrole integrity dat, indexování v databázích, prokazování znalosti, apod.

Kapitola 1

Konstrukce hašovacích funkcí

Všechny moderní používané hašovací funkce jsou konstruovány podle *Merkle-Dåmgardovy* konstrukce, kterou v této kapitole podrobně popíšeme. Tento způsob konstrukce navrhli a popsali Ralph Merkle a Ivan Dåmgard nezávisle na sobě na konferenci Crypto 1989.

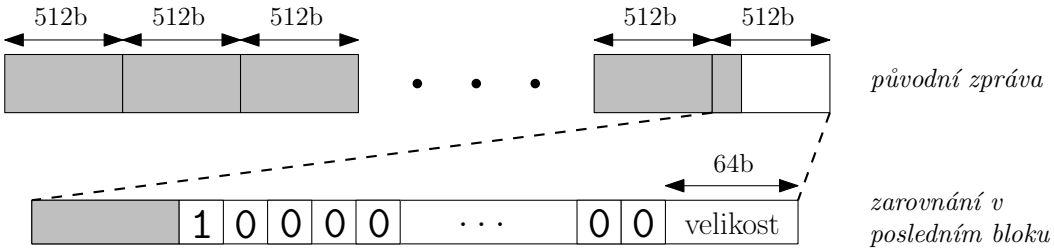
1.1 Zarovnání zprávy

Každá hašovací funkce musí být schopna zpracovat zprávu libovolné délky. Toho se dosáhne tím, že se zpráva nejprve rozdělí na stejně velké bloky a poté je po těchto blocích postupně zpracovávána. Proto je potřeba nejprve zprávu zarovnat, aby se dala rozdělit na bloky o stejné (předem dané) velikosti. Způsob zarovnání musí umožňovat jednoznačné odejmutí, aby nevznikaly jednoduché kolize. Kdybychom například zprávu doplnili pouze nulovými bity, později už bychom nevěděli, které z těchto bitů jsou ještě platnými bity zprávy a které jsou ty doplněné. Zpráva se tedy zarovná nejprve přidáním bitu `1` a poté potřebným počtem bitů `0`. Při zarovnání tímto způsobem můžeme vždy jednoznačně doplněk odejmout. Pro větší bezpečnost se používá ještě tzv. *Merkle-Dåmgardovo zesílení*. To se provádí tak, že se navíc na konci zprávy rezervuje 64 bitů, do kterých se uloží velikost původní zprávy. Pokud potřebných 64 bitů není k dispozici, prodlouží se zpráva ještě o jeden blok. Na obrázku 1.1 je zarovnání ilustrováno pro nejobvyklejší délku bloku 512 bitů.

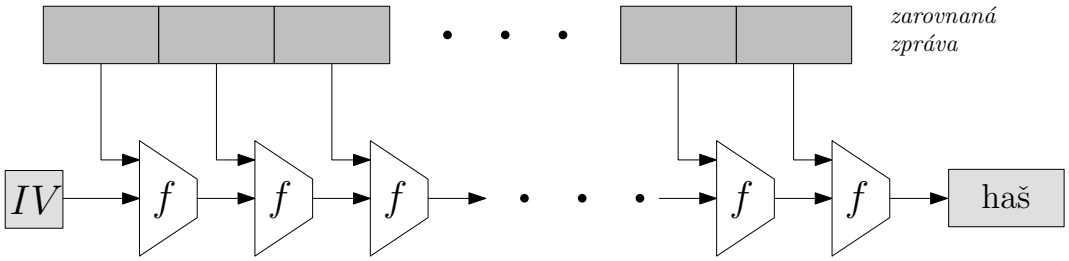
1.2 Kompresní funkce

Jednotlivé bloky se zpracovávají *kompresní funkcí*, označme ji f. Obrázek 1.2 znázorňuje postup zpracování zprávy. Na začátku do kompresní funkce vstupuje první blok zprávy a pevně stanovený *inicializační vektor* (IV), který je specifikován pro každou hašovací funkci. Výstup tvoří tzv. *kontext*. Tento kontext vstupuje do kompresní funkce spolu s dalším blokem zprávy v dalším kroku. Tímto způsobem se zpracují

všechny bloky zprávy. Kontext z posledního kroku se buď ještě nějakým způsobem upraví, nebo rovnou tvoří výslednou haš.



Obrázek 1.1: Zarovnání zprávy



Obrázek 1.2: Zpracování zprávy kompresní funkcí

1.3 Konstrukce MD5

Hašovací funkce MD5 byla navržena v roce 1992 Ronaldem Rivestem jako náhrada hašovací funkce MD4. Její detailní popis můžeme najít ve specifikaci RFC 1321 (viz [8]). Protože v této práci budeme využívat některé hodnoty vznikající během výpočtu, konstrukci MD5 si podrobně popíšeme.

Hašovací funkce MD5 rozděluje zprávu do bloků o velikosti 512 bitů. Kontext má velikost 128 bitů a v posledním kroku tvoří bez úpravy výslednou haš. Nyní k popisu hašovací funkce MD5 stačí popsat její kompresní funkci, která je definovaná v [8].

S kontextem se pracuje jako se čtyřmi 32-bitovými slovy[1]. Označme si kontext na vstupu jako I_1, I_2, I_3, I_4 a definujme

$$Q_{-3} = I_1, \qquad Q_{-2} = I_4, \qquad Q_{-1} = I_3, \qquad Q_0 = I_2.$$

Blok zprávy je rozdělen na šestnáct 32-bitových slov $m_0, m_1, \ldots, m_{15}$. Dále si definujme pomocné funkce potřebné k výpočtu. Na vstupu i výstupu jsou 32-bitová

[1] *Slovem* rozumíme binárně vyjádřené přirozené číslo nebo také vektor bitů.

slova a jednotlivé operace se provádějí po bitech ($\oplus$ značí bitové XOR).

$$F_i(X,Y,Z) = \begin{cases} (X \wedge Y) \vee (\neg X \wedge Z), & \text{pro } 1 \leq i \leq 16, \\ (X \wedge Z) \vee (Y \wedge \neg Z), & \text{pro } 17 \leq i \leq 32, \\ X \oplus Y \oplus Z, & \text{pro } 33 \leq i \leq 48, \\ Y \oplus (X \vee \neg Z), & \text{pro } 49 \leq i \leq 64. \end{cases}$$

Výpočet probíhá v 64 krocích, které jsou rozděleny do čtyř částí. V každé části se v 16 krocích zpracovává celý blok zprávy $m_0, m_1, \ldots, m_{15}$ pokaždé v jiné permutaci. Tyto čtyři permutace si označme jako jednu funkci $p(i)$, která zobrazuje i na index úseku zprávy, který se zpracovává v i-tém kroku.

$$p(i) = \begin{cases} i - 1, & \text{pro } 1 \leq i \leq 16, \\ 12 + 5i \bmod 16, & \text{pro } 17 \leq i \leq 32, \\ 2 + 3i \bmod 16, & \text{pro } 33 \leq i \leq 48, \\ 9 + 7i \bmod 16, & \text{pro } 49 \leq i \leq 64. \end{cases}$$

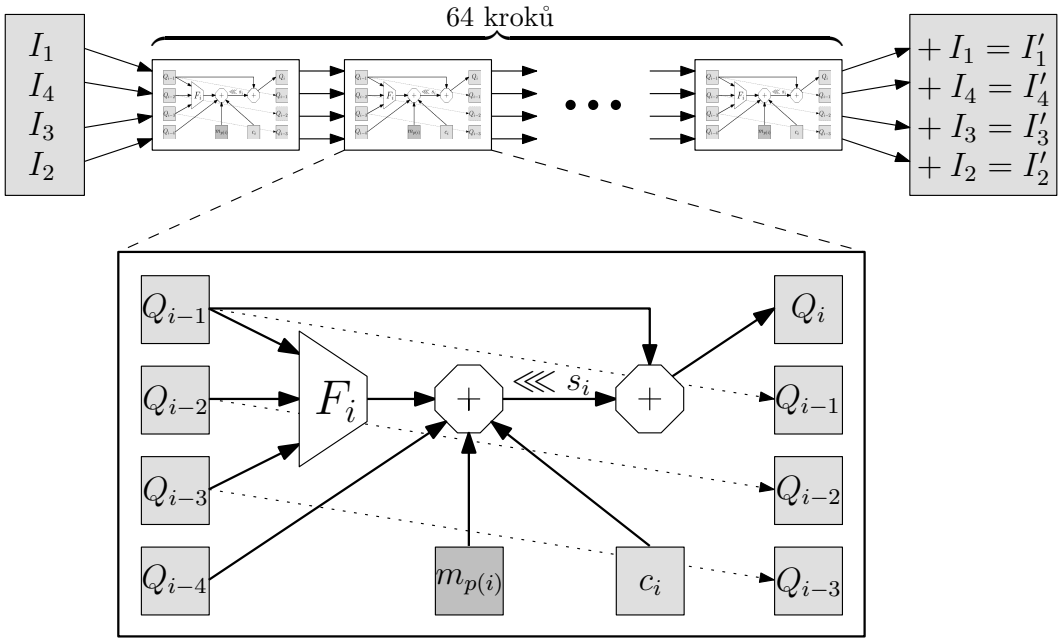
V každém kroku je zapotřebí konstanta c_i o maximální velikosti 32 bitů a konstanta s_i, která nabývá hodnot mezi 0 a 31 (všechny definované v [8]). Symbol $\lll x$ znamená cyklický posun o x bitů doleva. Dále v textu jsou všechny binární operace mezi 32-bitovými slovy chápány modulo 2^{32}. Nyní si již můžeme uvést výpočet, který probíhá v každém kroku $i = 1, 2, \ldots, 64$,

$$Z_i = F_i(Q_{i-1}, Q_{i-2}, Q_{i-3}) + Q_{i-4} + m_{p(i)} + c_i, \tag{1.1}$$
$$Q_i = Q_{i-1} + Z_i \lll s_i. \tag{1.2}$$

Hodnoty Q_i se nazývají *registry*. Výstup kompresní funkce tvoří registry Q_{61}, Q_{62}, Q_{63}, Q_{64}, ke kterým se přičte kontext ze vstupu,

$$\begin{aligned} I'_1 &= I_1 + Q_{61}, \\ I'_2 &= I_2 + Q_{64}, \\ I'_3 &= I_3 + Q_{63}, \\ I'_4 &= I_4 + Q_{62}. \end{aligned}$$

Hodnoty I'_1, I'_2, I'_3, I'_4 vstupují do dalšího výpočtu, nebo v případě, kdy se zpracoval poslední blok zprávy, jsou brány jako výsledná haš. Na obrázku 1.3 si můžeme prohlédnou ilustraci průběhu jednotlivých kroků.



Obrázek 1.3: Kompresní funkce MD5

Kapitola 2

Statistické testy MD5

Jednou z požadovaných vlastností hašovacích funkcí je náhodný výstup. V této kapitole uvedeme jednoduché statistické testy, pomocí kterých náhodnost výstupu hašovací funkce MD5 ověříme.

Jako vstup hašovací funkce vždy volíme náhodnou posloupnost 512 bitů a pak analyzujeme výstup. Budeme tudíž potřebovat nějaký generátor (pseudo)náhodných bitů. Abychom odhalili případné nedostatky generátoru, budeme používat tři různé generátory. Nejprve si tyto generátory, které jsou použity ve všech testech, uvedeme a poté se budeme zabývat jednotlivými testy.

2.1 Generátory náhodných bitů

Jako první generátor náhodných bitů použijeme generátor náhodných čísel, který je standardně implementován v jazyce C, přičemž upravíme jeho výstup na požadovanou velikost v bitech. Označme tento generátor G_1 a jeho výstup $G_1(s)$, kde s určuje velikost výstupu v bitech.

Jako druhý generátor použijeme volně šiřitelný generátor Mersenne Twister (viz [7]). Označme ho G_2 a jeho výstup $G_2(s)$.

Třetí generátor G_3 zkonstruujeme podle následujícího algoritmu.

Algoritmus 2.1 (Generátor náhodných bitů G_3)**.**
`VSTUP:` s.
`VÝSTUP:` $G_3(s)$.

1. $RN := G_1(24)$.
2. $c := 2G_1(12) + 1 \bmod 2^{12}$. } inicializace
3. RN ulož do výstupního pole bez úvodních nulových bitů a prvního bitu 1.
4. $RN := (RN + c) \bmod 2^{24}$.
5. Pokud výstupní pole nemá velikost aspoň s bitů, jdi na krok 3.
6. `ODPOVĚZ` prvních s bitů z výstupního pole a odeber je.

Při prvním volání generátoru G_3 se v 1. a 2. kroku generátor inicializuje. Při každém dalším volání se inicializace již neprovádí a začíná se krokem 3. Uveďme příklad, který nám generátor G_3 více přiblíží.

Příklad 2.2. Proveďme jednotlivé kroky algoritmu pro vstup $s = 32$.

- Nechť je slovo RN například `001011101001011111011101` (24 bitů) a slovo $c =$ `000100110101` (12 bitů).
- Uložme do výstupního pole slovo RN bez prvních tří bitů `001`. Takže máme

  `VÝSTUPNÍ POLE: 01110100101111011101` (20 bitů).
- Velikost výstupního pole je menší než 32 bitů, pokračujme krokem 3.
- Upravme RN na `001011101001101000010010` (24 bitů).
- Ze slova RN odeberme opět první tři bity a přidejme ho do výstupního pole

  `VÝSTUPNÍ POLE: 0111010010111101110101110100110100010010` (40 bitů).
- Velikost výstupního pole již přesáhla velikost 32 bitů, pokračujme krokem 6.
- Výstup z algoritmu tvoří řetězec

  $G_3(32) =$ `01110100101111011101011101001101` (32 bitů).
- Při dalším volání generátoru se již neprovádí inicializace a začíná se s neprázdným výstupním polem

  `VÝSTUPNÍ POLE: 00010010` (8 bitů).

V testech z kapitoly 2 je navíc ošetřeno zacyklení generátoru po 32-bitových slovech (v těchto testech se generuje pokaždé 512 bitů). Každých 32 bitů se zkontroluje, zda není prvních 32 vygenerovaných bitů rovno posledním 32 bitům. Pokud ano, tak se vygeneruje ještě dalších 15×32 bitů, které se porovnají s druhým až šestnáctým 32-bitovým slovem od začátku. Pokud dojde ke shodě, výpočet pokračuje s nově inicializovaným generátorem (tolerovány jsou maximálně 2 zacyklení).

2.2 Test náhodnosti bitů

Prvním testem se pokusíme zjistit, zda se na i-té pozici v haši náhodné zprávy vyskytne bit `1` se stejnou pravděpodobností jako bit `0`. Budeme potřebovat některé poznatky ze základů teorie pravděpodobnosti a matematické statistiky (viz [2]).

2.2.1 Použitá teorie

Nechť $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{in})^T, i = 1, 2, \ldots, u$ jsou náhodné výběry z alternativního rozdělení s parametrem p_i. Pro každé $i = 1, 2, \ldots, u$ budeme testovat hypotézu, že $p_i = p_i^0$, kde $p_i^0 \in (0, 1)$. Podle tab. 6.4 z [2] tuto hypotézu pro velká n zamítáme na asymptotické hladině α (pravděpodobnost chyby prvního druhu[1]), pokud platí nerovnost

$$\frac{|\overline{X}_{in} - p_i^0|}{\sqrt{p_i^0(1 - p_i^0)}}\sqrt{n} \geq u_{1-\alpha/2}, \tag{2.1}$$

kde u_β je β-kvantil normovaného normálního rozdělení, který je tabelován, a $\overline{X}_{in}$ je výběrový průměr i-tého náhodného výběru definovaný vzorcem

$$\overline{X}_{in} = \frac{1}{n}\sum_{k=1}^{n} X_{ik}.$$

Označme

$$n_i = n\overline{X}_{in}$$

a

$$A_i = \left(np_i^0 - u_{1-\alpha/2}\sqrt{np_i^0(1 - p_i^0)}, \quad np_i^0 + u_{1-\alpha/2}\sqrt{np_i^0(1 - p_i^0)}\right). \tag{2.2}$$

Místo nerovnosti (2.1) můžeme ekvivalentně ověřit, jestli platí

$$n_i \notin A_i.$$

2.2.2 Algoritmus

Na základě odstavce 2.2.1 můžeme sestavit první algoritmus na testování náhodnosti výstupu hašovací funkce MD5. Nechť hodnota i-tého bitu haše je náhodná veličina s alternativním rozdělením o parametru p_i. Pro každé $i = 1, 2, \ldots, 128$ budeme testovat, zda $p_i = 0,5$. Postupně vygenerujeme n náhodných zpráv, které zhašujeme. Tím získáme náhodný výběr, který můžeme testovat způsobem uvedeným v odstavci 2.2.1 pro $u = 128$. Nyní již zformulujeme konkrétní algoritmus, který bude test realizovat.

Algoritmus 2.3 (Test náhodnosti bitů)**.**
`VSTUP`: $\emptyset$.
`VÝSTUP`: Pole výsledků $(t_1, t_2, \ldots, t_{128})$.

[1] Chyby prvního druhu se dopustíme, zamítneme-li pravdivou hypotézu.

1. Pro $k = 1, 2, \ldots, n$ opakuj:
   - $(X_{1k}, X_{2k}, \ldots X_{128k}) := \mathcal{MD}5(G_r(512))$.
2. Pro $i = 1, 2, \ldots, 128$ opakuj:
   - $n_i := \sum_{k=1}^{n} X_{ik}$.
   - Pomocí (2.2) vyjádři interval A_i.
3. Do pole $(t_1, t_2, \ldots, t_{128})$ ulož výsledek testování

$$t_i := \begin{cases} 1, & \text{když } n_i \notin A_i, \\ 0, & \text{když } n_i \in A_i. \end{cases}$$

4. `ODPOVĚZ` $(t_1, t_2, \ldots, t_{128})$.

2.2.3 Vyhodnocení

Test byl naprogramován podle algoritmu 2.3 s konkrétními hodnotami $n = 10^4$ a $\alpha = 0,05$, resp. $\alpha = 0,01$. Aby byla odhalena chyba způsobená zvolenou hladinou testování, byl test opakován 100krát. Výsledky najdeme v dodatku v tabulkách A.1 a A.2. Každá buňka odpovídá počtu zamítnutí hypotézy na i-té pozici při použití jednoho z generátorů. Analyzováním výsledků zjistíme, že na obou hladinách jsou četnosti zamítnutí pro všechny pozice poměrně malé, tudíž tyto případy nemají žádný statisticky význam. Můžeme prohlásit, že podle použité metody testování je pravděpodobnost výskytu bitu `1` na jakékoliv pozici haše náhodné zprávy stejná jako pravděpodobnost výskytu bitu `0`.

2.3 Test nezávislosti bitů

V dalším testu se budeme zabývat nezávislostí jednotlivých bitů výstupu. Budeme testovat, zda jsou bity po dvou nezávislé. Použitou teorii ([1], kap.13 - Kontingenční tabulky) si opět nejprve shrneme.

2.3.1 Použitá teorie

Nechť $(\boldsymbol{X}_i, \boldsymbol{X}_j)$, $i, j = 1, 2, \ldots, u$ je náhodný výběr ve smyslu odstavce 2.2.1. Pro dané i a j si označme n_{ab} počet těch případů, kdy se ve výběru vyskytla dvojice (a, b), $a, b \in \{0, 1\}$. Matice (n_{ab}) se nazývá *čtyřpolní kontingenční tabulka*. Dále definujme

$$n_{a.} = n_{a0} + n_{a1} \quad \text{a} \quad n_{.b} = n_{0b} + n_{1b}. \tag{2.3}$$

Zřejmě platí

$$n = n_{0.} + n_{1.} = n_{.0} + n_{.1} = n_{00} + n_{01} + n_{10} + n_{11}.$$

Podle věty 13.3 z [1] zamítáme hypotézu, že odpovídající náhodné veličiny jsou nezávislé, právě když platí

$$\frac{n_{0.}n_{1.}n}{n_{.0}n_{.1}}\left(\frac{n_{00}}{n_{0.}}-\frac{n_{10}}{n_{1.}}\right)^2 \geq \chi^2_{1-\alpha,1}, \tag{2.4}$$

kde $\chi^2_{\beta,1}$ je β-kvantil χ^2 rozdělení o jednom stupni volnosti, který je tabelován.

2.3.2 Algoritmus

Nyní můžeme zkonstruovat algoritmus, který bude test provádět. Situace je obdobná jako v odstavci 2.2.2. Nechť hodnoty bitů na i-té a j-té pozici haše náhodné zprávy jsou náhodné veličiny. Vygenerujeme n náhodných zpráv, které zhašujeme a tím získáme náhodné výběry, pomocí kterých budeme podle odstavce 2.3.1 pro $u = 128$ testovat nezávislost těchto dvou náhodných veličin. Pro potřebu algoritmu přeznačme n_{ab} na n^{ij}_{ab}, aby bylo patrné, k jaké dvojici bitů patří, a ještě definujme pomocnou funkci

$$\Phi(ab, X, Y) := \begin{cases} 1, & \text{když } X = a \text{ a } Y = b, \\ 0, & \text{jinak.} \end{cases} \tag{2.5}$$

Algoritmus 2.4 (Test nezávislosti bitů)**.**
`VSTUP`: $\emptyset$.
`VÝSTUP`: Matice výsledků (t_{ij}), $i, j = 1, 2, \ldots, 128$.

1. Pro $k = 1, 2, \ldots, n$ opakuj:
   - $(X_{1k}, X_{2k}, \ldots, X_{128k}) := \mathcal{MD}5(G_r(512))$.
2. Pro všechny dvojice (i, j), $j > i$, $i, j = 1, 2, \ldots, 128$ opakuj:
   - $n^{ij}_{00} := \sum_{k=1}^{n} \Phi(00, X_{ik}, X_{jk})$,
   - $n^{ij}_{01} := \sum_{k=1}^{n} \Phi(01, X_{ik}, X_{jk})$,
   - $n^{ij}_{10} := \sum_{k=1}^{n} \Phi(10, X_{ik}, X_{jk})$,
   - $n^{ij}_{11} := \sum_{k=1}^{n} \Phi(11, X_{ik}, X_{jk})$.
   - Dopočti $n^{ij}_{.0}, n^{ij}_{.1}, n^{ij}_{0.}, n^{ij}_{1.}$ podle (2.3).
   - Spočti výraz z (2.4) a ulož ho do γ_{ij}.
3. Do matice (t_{ij}) ulož výsledek testování

$$t_{ij} := \begin{cases} 1, & \text{když } \gamma_{ij} \geq \chi^2_{1-\alpha,1}, \\ 0, & \text{jinak.} \end{cases}$$

4. `ODPOVĚZ` (t_{ij}).

2.3.3 Vyhodnocení

Podle algoritmu 2.4 byl test naprogramován pro konkrétní hodnoty $n = 10^4$ a $\alpha = 0,05$, resp. $\alpha = 0,01$. Test byl opakován opět 100krát na obou hladinách a pro

všechny tři generátory. Protože velikost tabulky neumožňuje přiložit ji v tištěné podobě, je přiložena na CD spolu se zdrojovým kódem programu. Výsledné hodnoty jsou přibližně stejné jako v předchozím testu, takže opět můžeme prohlásit, že podle použité metody jsou bity výstupu hašovací funkce po dvou nezávislé.

2.4 Test senzitivity

V třetím testu se pokusíme zjistit, jaký vliv má na výslednou haš změna jednoho bitu ve zprávě. Tentokrát žádné statistické metody používat nebudeme. Jednoduše budeme sledovat počet změněných bitů v haších náhodných zpráv při změně jednoho bitu a z výsledků vyhotovíme graf.

2.4.1 Algoritmus

Algoritmus sestavíme tak, aby v cyklu měnil v náhodné zprávě vždy jeden z 512 bitů a do výsledného pole ukládal, v kolika případech se v haši změnilo právě c bitů. To budeme opakovat pro n náhodných zpráv.

Algoritmus 2.5 (Test senzitivity)**.**

`VSTUP`: $\emptyset$.
`VÝSTUP`: Pole výsledků $(t_0, t_1, \ldots, t_{128})$.

1. Pro $k = 1, 2, \ldots, n$ opakuj:
   - $\boldsymbol{M} = (M_1, M_2, \ldots, M_{512}) := G_r(512)$,
   - $\boldsymbol{X} = (X_1, X_2, \ldots, X_{128}) := \mathcal{MD}5(\boldsymbol{M})$.
   - Pro $l = 1, 2, \ldots, 512$ opakuj:
     - $\boldsymbol{M'} := (M_1, \ldots, \neg M_l, \ldots, M_{512})$,
     - $\boldsymbol{X'} := \mathcal{MD}5(\boldsymbol{M'})$,
     - $\boldsymbol{Y} := \boldsymbol{X} \oplus \boldsymbol{X'}$,
     - $c := \sum_{i=1}^{128} Y_i$,
     - $t_c := t_c + 1$.
2. `ODPOVĚZ` $(t_0, t_1, \ldots, t_{128})$.

2.4.2 Vyhodnocení

Test byl naprogramován podle algoritmu 2.5 pro $n = 10^5$. Výsledek najdeme v tabulce A.3. První tři sloupce odpovídají výsledkům podle použitého generátoru a v posledním sloupci je jejich součet, z něhož je vyhotoven graf, který najdeme na obrázku A.1. Z grafu je patrné, že případ, kdy došlo ke změně přesně poloviny bitů v haši, je zastoupen nejvíce. Četnost případů, kdy se změnilo bitů více nebo méně, si navzájem symetricky odpovídá a má klesající tendenci.

Pro zajímavost byl proveden ještě jeden test, v kterém se sledují zprávy, v jejichž haši došlo k nejmenší, nebo naopak k největší změně. Test běžel dva dny a zkontroloval přibližně 21×10^8 náhodných zpráv (použity všechny tři generátory).

Maximální změna
počet bitů: 103, na pozici: 425, zpráva[2]:
`0xa0a96840,0x541e7eaa,0xd8f95ae9,0xeac043c5,0x95676b54,0x8e96fc9c,`
`0xaea0385e,0x8e6ebfb8,0xd984a9ab,0x13c3463c,0x666ca28c,0x999898b3,`
`0xd83068fe,0x5ed49ab9,0xae696b67,0x42c6e365.`

Minimální změna
počet bitů: 27, na pozici: 492, zpráva:
`0x585408b0,0xfed33ede,0x8f4109e0,0xc83b2de7,0x4858e55e,0x9c364378,`
`0x166d2c8d,0x6ac9a600,0x33d4e65d,0x38c7fa98,0x04e178a1,0x970f5169,`
`0x489619c1,0x59ef9524,0x314a9694,0x2ea12b3d.`

Vidíme, že by tato metoda byla na hledání kolizí nevhodná. Úspěšným metodám hledání kolizí se budeme věnovat v další kapitole.

[2]Zpráva je zapsána po 32-bitových slovech, která jsou vyjádřena hexadecimálně.

Kapitola 3

Princip útoků na MD5

Hašovací funkce MD5 byla doposud předmětem mnoha kryptoanalytických útoků. My si uvedeme nejzásadnější útoky z poslední doby, které popřely její bezpečnost z hlediska bezkoliznosti.

Poprvé byla uveřejněna úplná kolize MD5 na konferenci Crypto 2004 v srpnu roku 2004 čínským týmem Wangové (viz [10]). Ještě předtím, než tým Wangové uveřejnil metodu hledání kolizí (viz [11]), se jí podařilo rekonstruovat Vlastimilu Klímovi (viz [3, 4]). Princip útoku si nyní popíšeme.

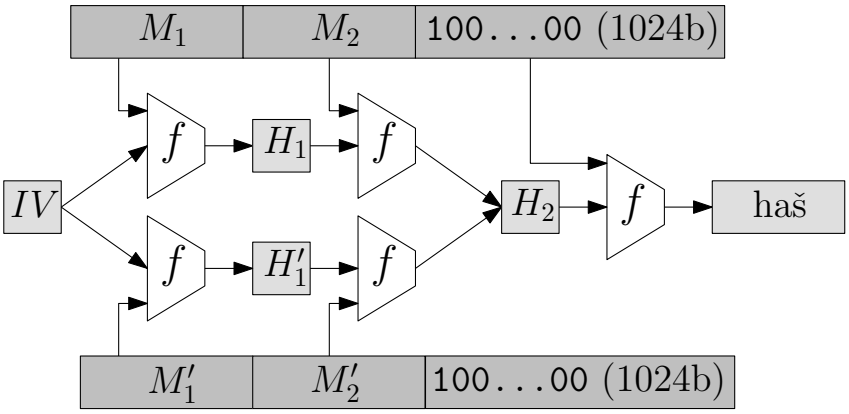
3.1 Čínský útok

Kolidující zprávy se skládají ze dvou bloků. Označme bloky první zprávy M_1, M_2 a druhé zprávy M_1', M_2'. První a druhé bloky zpráv se liší o předem stanovený vektor C_1 a C_2. Pokud jsou při zpracování prvního bloku M_1 kompresní funkcí splněny určité postačující podmínky, je zaručeno, že se liší i kontexty $H_1 = f(IV, M_1)$ a $H_1' = f(IV, M_1')$ o předem daný vektor C_3. Když při zpracování druhého bloku M_2 opět splníme určité podmínky, výsledné kontexty $H_2 = f(H_1, M_2)$ a $H_2' = f(H_1', M_2')$ se rovnají. Takže platí

$$M_1' = M_1 + C_1, \quad M_2' = M_2 + C_2, \quad H_1' = H_1 + C_3, \quad H_2' = H_2.$$

Při hašování se obě zprávy na začátku doplní o blok, který je v obou případech stejný, jelikož zprávy mají stejnou velikost 1024 bitů. Takže z rovností kontextů H_2' a H_2 plyne i rovnost výsledné haše. Na obrázku 3.1 je popsaný útok ilustrován.

Výhodou je, že tento způsob hledání kolizí funguje pro libovolný inicializační vektor IV, což nám umožňuje kolidující zprávu navazovat na libovolnou smysluplnou zprávu. Toho se dá využít při mnoha útocích v praxi, ale toto téma je již mimo rozsah této práce. My se budeme nadále věnovat oněm postačujícím podmínkám, které je potřeba splnit při tomto útoku.



Obrázek 3.1: Princip čínského útoku

3.2 Postačující podmínky

Vektory C_1, C_2 a C_3, určující diference mezi zprávami a mezi prvními kontexty, nabývají hodnot

$$\begin{aligned} C_1 &= (0,0,0,0,2^{31},0,0,0,0,0,0,2^{15},0,0,2^{31},0), \\ C_2 &= (0,0,0,0,2^{31},0,0,0,0,0,0,-2^{15},0,0,2^{31},0), \\ C_3 &= (2^{31},2^{31}+2^{25},2^{31}+2^{25},2^{31}+2^{25}). \end{aligned}$$

Z těchto diferencí tým Wangové odvodil tzv. *diferenční cestu*, což jsou diference mezi jednotlivými registry (viz (1.2)), které vznikají při zpracovávání prvního a druhého bloku zpráv (M_1, M_2) a (M'_1, M'_2). Tyto diference mezi registry jsou jak ve smyslu rozdílu modulo 2^{32}, tak ve smyslu bitového XOR. Diferenční cesta je uvedená v tabulkách 3 a 5 v [11], kde modulární diferenci odpovídá šestý sloupec a XORové sedmý. Postačující podmínky vyplývají právě z této diferenční cesty a jsou kladeny přímo na bity registrů vzniklých při zpracování (M_1, M_2). Způsob odvozování podmínek je popsán v [11], kde jsou i tabulky s podmínkami tab. 4 a 6. Tyto podmínky byly shledány jako neúplné a některé z nich byly dokonce chybné. Byly opraveny například v práci [6], z které je převzal a využil Vlastimil Klíma k vytvoření zatím nejrychlejšího algoritmu na hledání kolizí (viz [5]). Opraveny byly také Marcem Stevensem v jeho práci [9], v které mimo jiné popisuje algoritmus svého útoku. Těmito dvěma algoritmy se nadále budeme zabývat. Jelikož naplnění podmínek při zpracovávání druhého bloku je jednodušší než u prvního bloku, budeme se věnovat pouze zpracování prvního bloku zprávy M_1.

3.3 Stevensův útok

Marc Stevens ve své práci [9] opravil a odvodil chybějící postačující podmínky k naplnění diferenční cesty z [11] (viz tabulky v dodatku A.4, A.5) a také popsal algoritmus, který pomocí těchto podmínek hledá kolize. Vychází z původní myšlenky, která byla použita už v práci Wangové a kol. [11] a Klímy [3, 4], tzv. *mnohonásobné modifikace zprávy*. Nejprve se náhodně zvolí několik počátečních registrů splňujících předepsané podmínky tak, aby bylo ještě možné jednoznačně dopočítat první blok zprávy. Pro úplnost uveďme, že část zprávy $m_{p(i)}$ lze z rovnic (1.1) a (1.2) vyjádřit

$$m_{p(i)} = (Q_i - Q_{i-1}) \ggg s_i - F_i(Q_{i-1}, Q_{i-2}, Q_{i-3}) - Q_{i-4} - c_i.$$

Když je blok zprávy již pevně stanoven, nezbývá nic jiného než dopočítat ostatní registry a ověřit, jestli splňují zbylé postačující podmínky. Ty už jsou splněny pouze pravděpodobnostně[1]. V případě, že splněny nejsou, pozmění se bity některých počátečních registrů tak, aby stále splňovaly podmínky. Změna těchto registrů se projeví na zprávě a tím pádem také na zbylých registrech. Poté se opět ověří podmínky. Tímto způsobem po určitém počtu kroků dochází k naplnění všech podmínek. Nyní si Stevensův algoritmus uvedeme v konkrétní podobě, jak ho popsal ve své práci [9].

Algoritmus 3.1 (Stevensův)**.**
`VSTUP`: $\emptyset$.
`VÝSTUP`: První blok zprávy $(m_0, m_1, \ldots, m_{15})$.

1. Náhodně zvol $Q_1, Q_3, \ldots, Q_{16}$, aby splňovaly podmínky (tab. A.4, A.5).
2. Spočti $m_0, m_6, \ldots, m_{15}$.
3. Dokud $Q_{18}, \ldots, Q_{21}$ nesplňuje podmínky, opakuj:
   - Náhodně zvol Q_{17}, aby splňoval podmínky.
   - Spočti m_1 z $Q_{13}, Q_{14}, Q_{15}, Q_{16}, Q_{17}$.
   - Spočti Q_2 a m_2, m_3, m_4, m_5.
   - Spočti $Q_{18}, \ldots, Q_{21}$.
4. Pro všechny možné Q_9, Q_{10} splňující podmínky tak, aby se m_{11} nezměnilo, opakuj:
   - Spočti $m_{10}, Q_{22}, Z_{23}, Q_{23}, Q_{24}, m_8, m_9, m_{12}, m_{13}$.
   - Spočti $Q_{25}, \ldots, Q_{34}, Z_{35}, Q_{35}, \ldots, Q_{64}, I'_2, I'_3, I'_4$.
   - Ověř zbylé podmínky. Pokud jsou splněny, zastav a `ODPOVĚZ` $(m_0, m_1, \ldots, m_{15})$.
5. Jdi na krok 1.

3.4 Klímův útok

Vlastimil Klíma ve své práci [5] přišel s novým nápadem, který několikanásobně urychlil jeho původní algoritmus (viz jeho předchozí publikace [3, 4]). Nalezení jedné

[1] Tím myslíme s pravděpodobností menší než 1.

kolize trvá v průměru již jen několik desítek sekund místo původních několika hodin. Jeho algoritmus je opět postaven na principu mnohonásobné modifikace zprávy, ale navíc využívá tzv. *tunely*. Na začátku se klasicky naplní podmínky pro počáteční registry, spočte se zpráva a když už nezbývá nic jiného, než spočítat zbývající registry a ověřit, jestli splňují předepsané podmínky, přicházejí na řadu tunely. Tunelem Klíma nazývá sadu bitů nějakého registru, které lze měnit bez vlivu na již splněné podmínky. Tato změna se vždy projeví na některých hodnotách m_i a tím pádem také na zbylých registrech, které splňují postačující podmínky pouze pravděpodobnostně. Tímto způsobem se z jednoho kandidáta na splnění všech podmínek „vytuneluje“ kandidátů hned několik a tím se zvyšuje šance na úspěch. Tunel nazývá podle registru, v kterém můžeme bity měnit a ještě definuje *sílu tunelu*, což je počet těchto bitů. Pomocí tunelu o síle k lze rozmnožit jednoho kandidáta na 2^k kandidátů. Pokud máme ještě jiný tunel o síle l, můžeme každého takového kandidáta opět rozmnožit na dalších 2^l, takže už máme dohromady 2^{k+l} kandidátů z jednoho původního. Vidíme, že síla tunelů se sčítá. Klíma využívá ve svém algoritmu šest tunelů, konkrétně tunely $\boldsymbol{Q}_4$, $\boldsymbol{Q}_9$, $\boldsymbol{Q}_{10}$, $\boldsymbol{Q}_{13}$, $\boldsymbol{Q}_{14}$, $\boldsymbol{Q}_{20}$ (jejich popis najdeme v [5]). Některé využívají tzv. *extra podmínky*, jež jsou kladeny na registry $Q_4, \ldots, Q_{16}$. Používá je spolu s převzatými postačujícími podmínkami z [6] a najdeme je v dodatku v tabulkách A.6 a A.7. Tunely se dělí na *deterministické*, *pravděpodobnostní* a *dynamické*. Deterministické tunely pokaždé zachovají všechny již splněné podmínky a umožní nám vytvořit pevný počet nových kandidátů. Pravděpodobnostní tunely nám splněné podmínky zachovají jen s určitou pravděpodobností (podle toho se také odvozuje síla takových tunelů) a dynamické tunely pokaždé volí postup tunelování podle daného stavu registrů. Nyní si uvedeme Klímův algoritmus (algoritmus je popsán pouze přibližně, v [5] je odkaz na zdrojový kód, z kterého lze vyčíst přesný algoritmus).

Algoritmus 3.2 (Klímův)**.**
`VSTUP`: $\emptyset$.
`VÝSTUP`: První blok zprávy $(m_0, m_1, \ldots, m_{15})$.

1. Náhodně zvol $Q_1, Q_3, \ldots, Q_{17}$, aby splňovaly podmínky (tab. A.6, A.7).
2. Spočti m_1, Q_2.
3. Spočti $m_0, m_4, m_5, m_6, m_{10}, m_{11}, m_{15}$.
4. Spočti $Q_{18}, Z_{19}, Q_{19}, Z_{20}, Q_{20}, Q_{21}, Q_{22}, Z_{23}, Q_{23}, Q_{24}$.
5. Pokud hodnoty z předchozího kroku nesplňují podmínky, jdi na krok 1.
6. Přes všechny možnosti tunelu $\boldsymbol{Q}_{10}$ opakuj:
    - Uprav a dopočti potřebné hodnoty.
    - Přes všechny možnosti tunelu $\boldsymbol{Q}_{20}$ opakuj:
        - Uprav a dopočti potřebné hodnoty.
        - Přes všechny možnosti tunelu $\boldsymbol{Q}_{13}$ opakuj:
            - Uprav a dopočti potřebné hodnoty.
            - Přes všechny možnosti tunelu $\boldsymbol{Q}_{14}$ opakuj:
                - Uprav a dopočti potřebné hodnoty.
                - Přes všechny možnosti tunelu $\boldsymbol{Q}_{4}$ opakuj:
                    - Uprav a dopočti potřebné hodnoty.
                    - Přes všechny možnosti tunelu $\boldsymbol{Q}_{9}$ opakuj:
                        - Uprav a dopočti potřebné hodnoty.
                        - Ověř zbylé podmínky. Pokud jsou splněny, zastav a `ODPOVĚZ` $m_0, m_1, \ldots, m_{15}$.
7. Jdi na krok 1.

Kapitola 4

Statistické testy postačujících podmínek

V této kapitole uvedeme testy, pomocí kterých budeme zjišťovat, zda postačující podmínky z předchozí kapitoly, které jsou splněny pouze pravděpodobnostně, jsou splněny náhodně. Tím se myslí skutečnost, že je pravděpodobnost splnění každé z nich $0,5$ a že jsou po dvou nezávislé. Budeme testovat podmínky ze Stevensova (tab. A.4, A.5) a Klímova (tab. A.6, A.7) algoritmu. Používat budeme generátory náhodných bitů G_1, G_2, G_3 popsané v kapitole 2.

4.1 Test náhodnosti splnění podmínek

Budeme testovat všechny podmínky z algoritmů Klímy a Stevense vyjma podmínek kladených na registry Q_3, $Q_4, \ldots,$ Q_{17}, které můžeme splnit s pravděpodobností 1. Označme testované podmínky ze Stevensova algoritmu $S1$, $S2, \ldots,$ $S38$ (viz tab. A.4, A.5) a z Klímova algoritmu $K1, K2, \ldots, K39$ (viz tab. A.6, A.7). Počet testovaných podmínek z Klímova, resp. Stevensova algoritmu označme jako v.

4.1.1 Algoritmus

Algoritmus testování založíme na teorii z odstavce 2.2.1 pro $u = v$. Splnění či nesplnění i-té podmínky představuje náhodnou veličinu s alternativním rozdělením o parametru p_i. Pro každé $i = 1, \ldots, v$ budeme testovat hypotézu, že platí $p_i = 0,5$. Výjimku budou tvořit podmínky $K4$ a $K7$, u kterých budeme testovat hypotézu, že $p_4 = (1 - 2^{-15})$, resp. $p_7 = (1 - 2^{-3})$, protože jsou to podmínky kladené na více bitů zároveň.

Algoritmus 4.1 (Test náhodnosti splnění podmínek)**.**
`VSTUP:` $\emptyset$.
`VÝSTUP:` Pole výsledků $(t_1, t_2, \ldots, t_v)$.

1. Pro $k = 1, 2, \ldots, n$ opakuj:
   - Náhodně zvol registry $Q_1, Q_3, \ldots, Q_{17}$, pomocí generátoru G_r tak, aby splňovaly podmínky.
   - Spočti m_1, Q_2.
   - Spočti $m_0, m_2, \ldots, m_{15}$.
   - Spočti $Q_{18}, \ldots, Q_{64}, Z_{19}, Z_{20}, Z_{23}, Z_{35}, I'_1, I'_2, I'_3$.
   - Pro $i = 1, \ldots, v$ opakuj:
     - Urči hodnotu X_{ik}:

$$X_{ik} := \begin{cases} \texttt{1}, & \text{když je splněna podmínka } Ki, \text{ resp. } Si, \\ \texttt{0}, & \text{jinak.} \end{cases}$$

2. Pro $i = 1, 2, \ldots, v$ opakuj:
   - $n_i := \sum_{k=1}^{n} X_{ik}$.
   - Pomocí (2.2) vyjádři interval A_i.
3. Do pole $(t_1, t_2, \ldots, t_v)$ ulož výsledek testování

$$t_i := \begin{cases} 1, & \text{když } n_i \notin A_i, \\ 0, & \text{když } n_i \in A_i. \end{cases}$$

4. `ODPOVĚZ` $(t_1, t_2, \ldots, t_v)$.

4.1.2 Vyhodnocení

Test byl proveden opět 100krát, pro $n = 10^4$ na hladinách $\alpha = 0,05$ a $\alpha = 0,01$. V tabulkách A.8 a A.9 jsou výsledky testování Stevensových podmínek a v tabulkách A.10 a A.11 jsou výsledky testování Klímových podmínek. Testované hypotézy zamítáme jen v několika málo případech. Proto můžeme učinit závěr, že skutečné pravděpodobnosti splnění jednotlivých podmínek se příliš neliší od očekávaných hodnot.

4.2 Test nezávislosti splnění podmínek

Zbývá nám už jen otestovat, jestli je splnění podmínek po dvou nezávislé. Značení převezmeme z předchozího testu. Jelikož četnost nesplnění podmínky $K4$ je příliš malá, nedovoluje nám to objektivně testovat její nezávislost s ostatními podmínkami, a proto ji vynecháme (v algoritmu se s ní pro jednoduchost počítá, ale na výstupu není).

4.2.1 Algoritmus

Použijeme teorii z odstavce 2.3.1 pro $u = v$. Budeme testovat hypotézu, že náhodné veličiny odpovídající splnění či nesplnění i-té a j-té podmínky jsou nezávislé. V algoritmu opět využijeme pomocnou funkci Φ definovanou v (2.5).

Algoritmus 4.2 (Test 5)**.**
`VSTUP`: $\emptyset$.
`VÝSTUP`: Matice výsledků (t_{ij}), $i, j = 1, 2, \ldots, v$.

1. Pro $k = 1, 2, \ldots, n$ opakuj:
   - Náhodně zvol registry $Q_1, Q_3, \ldots, Q_{17}$, pomocí generátoru G_r tak, aby splňovaly podmínky.
   - Spočti m_1, Q_2.
   - Spočti $m_0, m_2, \ldots, m_{15}$.
   - Spočti $Q_{18}, \ldots, Q_{64}, Z_{19}, Z_{20}, Z_{23}, Z_{35}, I'_2, I'_3, I'_4$.
   - Pro $i = 1, \ldots, v$ opakuj:
     - Urči hodnotu X_{ik}:

$$X_{ik} := \begin{cases} \texttt{1}, & \text{když je splněna podmínka } Ki, \text{ resp. } Si, \\ \texttt{0}, & \text{jinak.} \end{cases}$$

2. Pro všechny dvojice (i, j), $j > i$, $i, j = 1, 2, \ldots, v$ opakuj:
   - $n_{00}^{ij} := \sum_{i=1}^{n} \Phi(00, X_{ik}, X_{jk})$,
   - $n_{01}^{ij} := \sum_{i=1}^{n} \Phi(01, X_{ik}, X_{jk})$,
   - $n_{10}^{ij} := \sum_{i=1}^{n} \Phi(10, X_{ik}, X_{jk})$,
   - $n_{11}^{ij} := \sum_{i=1}^{n} \Phi(11, X_{ik}, X_{jk})$.
   - Dopočti $n_{.0}^{ij}, n_{.1}^{ij}, n_{0.}^{ij}, n_{1.}^{ij}$ podle (2.3).
   - Spočti výraz z (2.4) a ulož ho do γ_{ij}.
3. Do matice (t_{ij}) ulož výsledek testování

$$t_{ij} := \begin{cases} 1, & \text{když } \gamma_{ij} \geq \chi^2_{1-\alpha,1}, \\ 0, & \text{jinak.} \end{cases}$$

4. `ODPOVĚZ` (t_{ij}) (v případě Klímových podmínek vynech 4. sloupec a 4. řádek).

4.2.2 Vyhodnocení

I poslední test byl proveden celkem 100krát, pro $n = 10^4$ na hladině $\alpha = 0,05$, resp. $\alpha = 0,01$. Z prostorových důvodů jsou uvedeny pouze výsledky získané s použitím generátoru náhodných bitů G_2. Výsledky získané pomocí ostatních generátorů byly obdobné.

V tabulkách A.12 a A.13 jsou výsledky testů Stevensových podmínek. Všechny testy provedené na obou hladinách zamítají hypotézu, že splnění Stevensových podmínek $S1$ a $S4$ je nezávislé. Stejný výsledek jsme získali i při použití ostatních generátorů. Tyto dvě podmínky odpovídají 32. bitům registrů Q_{18} a Q_{19}, které mají být nulové. Navíc byl proveden výpočet, který zjišťoval relativní četnosti všech možných stavů dvojice podmínek $S1$ a $S4$. V tabulce 4.1 jsou uvedeny výsledky, z kterých plyne, že při splnění jedné z podmínek, je v 58 % případů splněna i podmínka druhá. Ostatní podmínky jsou podle použité metody dle výsledků po dvou nezávislé.

$S1$ nesplněna $S4$ nesplněna	$S1$ nesplněna $S4$ splněna	$S1$ splněna $S4$ nesplněna	$S1$ splněna $S4$ splněna
29 %	21 %	21 %	29 %

Tabulka 4.1: Relativní četnosti splnění/neslpnění podmínek $S1$ a $S4$

V tabulkách A.14 a A.15 jsou výsledky testování Klímových podmínek. Ani u podmínek Klímy nepotvrdily testy nezávislost ve všech případech. Nezávislost byla zamítnuta pro jednu dvojici podmínek opět ve všech provedených testech na obou hladinách pro všechny generátory. Jedná se o podmínky $K3$ a $K6$, které odpovídají 18. bitu registru Q_{18}, který má být `1` a 18. bitu Q_{19}, který má být `0`. Byl proveden stejný výpočet jako v předchozím odstavci. Výsledky najdeme v tabulce 4.2. Pokud je jedna z podmínek splněna, druhá je také splněna v 54,4 % případů. Ostatní podmínky se opět jeví jako po dvou nezávislé.

$K3$ nesplněna $K6$ nesplněna	$K3$ nesplněna $K6$ splněna	$K3$ splněna $K6$ nesplněna	$K3$ splněna $K6$ splněna
27,2 %	22,8 %	22,8 %	27,2 %

Tabulka 4.2: Relativní četnosti splnění/neslpnění podmínek $K3$ a $K6$

Je zajímavé, že závislé Klímovy podmínky $K3$ a $K6$ jsou zahrnuty i v Stevensových podmínkách pod označením $S3$ a $S5$ a jeví se jako nezávislé. To samé platí pro závislé Stevensovy podmínky $S1$ a $S4$. Obdobné Klímovy podmínky $K1$ a $K5$ jsou podle testů nezávislé. Zřejmě mají na závislost vliv podmínky na bity předchozích registrů, které mají oba mírně odlišné.

Závěr

V této práci jsme se zabývali statistickými testy hašovací funkce MD5. Testovali jsme, jestli tato hašovací funkce splňuje jeden ze základních požadavků, tedy zda je její výstup v jistém smyslu náhodný. Jednoduché testy nám hypotézu, že hašovací funkce MD5 tento požadavek splňuje, nevyvrátily. Uvedli jsme teorii, na které naše testování bylo založeno a podrobně jsme popsali použité algoritmy. Tyto algoritmy bychom mohli použít i na jiné hašovací funkce, ale to už by bylo nad rámec této práce.

Dále jsme si uvedli princip moderních útoků na MD5 a popsali jsme si konkrétní algoritmy na hledání kolizí Marca Stevense a Vlastimila Klímy. Podmínky, které je potřeba splnit při hledání kolizí v těchto dvou algoritmech, jsme podrobili statistickým testům. Pomocí testů jsme odhalili přímou závislost dvojic podmínek v obou algoritmech. Zajímavostí je, že tyto dvojice podmínek jsou v obou případech jiné.

Nakonec stojí za zmínku, že vzhledem ke zmíněným útokům na MD5 se doporučuje tuto funkci nadále nepoužívat a nahradit jí například některou z hašovacích funkcí SHA-2 nebo hašovací funkcí WHIRLPOOL.

Literatura

[1] Jiří Anděl: *Základy matematické statistiky*, Praha, 2005.

[2] Václav Dupač, Marie Hušková: *Pravděpodobnost a matematická statistika*, Praha, 2003.

[3] Vlastimil Klíma: *Finding MD5 Collisions – a Toy For a Notebook*, Cryptology ePrint Archive, Report 2005/075, Březen 2005, `http://eprint.iacr.org/2005/075.pdf`.

[4] Vlastimil Klíma: *Finding MD5 collisions on a notebook PC using multi-message modifications*, Cryptology ePrint Archive, Report 2005/102, Březen 2005, `http://eprint.iacr.org/2005/102.pdf`.

[5] Vlastimil Klíma: *Tunnels in Hash Functions: MD5 Collisions Within a Minute*, Cryptology ePrint Archive, Report 2006/105, Březen 2006, `http://eprint.iacr.org/2006/105.pdf`.

[6] Jie Liang, Xuejia Lai: *Improved Collision Attack on Hash Function MD5*, Cryptology ePrint Archive, Report 2005/425, Listopad 2005, `http://eprint.iacr.org/2005/425.pdf`.

[7] Makoto Matsumoto and Takuji Nishimura: *Mersenne Twister - A very fast random number generator*, `http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html`.

[8] Ronald Rivest: *The MD5 Message-Digest Algorithm*, RFC1321, Duben 1992, `http://rfc.net/rfc1321.html`.

[9] Marc Stevens: *Fast Collision Attack on MD5*, Cryptology ePrint Archive, Report 2006/104, Březen 2006, `http://eprint.iacr.org/2006/104.pdf`.

[10] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, Hongbo Yu: *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD*, Cryptology ePrint Archive, Report 2004/199, Srpen 2004, `http://eprint.iacr.org/2004/199.pdf`.

[11] Xiaoyun Wang, Hongbo Yu: *How to Break MD5 and Other Hash Functions*, Eurocrypt 2005, `http://www.infosec.sdu.edu.cn/paper/md5-attack.pdf`.

Dodatek A

Přílohy

A.1 Výsledky testu 2.2

i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
G_1	3	5	3	3	6	7	5	6	5	5	4	4	0	5	6	5	7	3	4	7	4	3	5	6	3	6	3	5	4	10	4	2
G_2	7	5	6	6	5	4	9	4	7	1	2	6	7	6	4	6	6	1	5	5	3	2	4	1	2	6	5	6	3	7	6	6
G_3	6	6	6	4	8	5	5	3	8	5	6	2	3	3	7	1	3	5	5	7	4	4	7	1	6	3	4	2	3	2	2	9

i	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**	**64**
G_1	5	8	6	5	5	4	2	4	6	5	5	8	7	5	1	2	0	6	7	4	7	2	8	3	6	6	7	2	5	4	1	5
G_2	7	5	3	3	4	3	6	4	3	5	5	4	6	1	5	7	6	4	3	5	6	5	3	5	3	3	2	2	5	3	4	3
G_3	6	8	8	5	4	3	2	3	5	5	7	2	9	4	6	6	4	4	6	3	5	7	7	4	6	4	6	7	3	3	6	4

i	**65**	**66**	**67**	**68**	**69**	**70**	**71**	**72**	**73**	**74**	**75**	**76**	**77**	**78**	**79**	**80**	**81**	**82**	**83**	**84**	**85**	**86**	**87**	**88**	**89**	**90**	**91**	**92**	**93**	**94**	**95**	**96**
G_1	5	9	3	4	3	2	7	9	7	4	3	7	5	3	6	4	7	2	7	6	1	8	6	7	7	3	4	2	8	4	4	3
G_2	8	4	4	4	6	4	1	5	4	5	5	5	6	1	7	5	6	4	9	5	2	4	5	3	7	5	4	4	8	4	6	8
G_3	4	3	4	5	5	3	6	9	2	7	4	6	4	4	1	5	6	7	3	7	3	4	6	6	11	6	4	4	5	9	4	4

i	**97**	**98**	**99**	**100**	**101**	**102**	**103**	**104**	**105**	**106**	**107**	**108**	**109**	**110**	**111**	**112**	**113**	**114**	**115**	**116**	**117**	**118**	**119**	**120**	**121**	**122**	**123**	**124**	**125**	**126**	**127**	**128**
G_1	11	9	7	6	7	5	6	4	10	5	3	8	8	5	8	2	5	7	8	4	3	7	11	6	6	5	9	6	2	2	7	8
G_2	6	5	5	8	7	5	3	8	8	5	3	3	4	7	2	5	7	3	4	1	4	10	3	2	4	2	2	8	5	6	5	4
G_3	4	6	8	7	4	7	3	8	3	6	2	3	7	5	5	10	6	8	4	5	6	8	6	4	2	1	6	6	10	4	3	4

Tabulka A.1: Výsledky testu 2.2 na hladině $0,05$

i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
G_1	1	1	3	0	2	2	0	1	2	0	0	0	2	0	1	2	1	1	2	1	1	0	0	1	2	2	0	1	1	0	1	0
G_2	0	1	1	2	0	0	0	0	1	3	4	3	4	1	2	1	2	0	1	2	1	0	1	1	0	1	1	0	3	1	3	1
G_3	0	0	1	0	2	1	2	2	2	1	4	1	2	0	1	3	1	0	0	2	1	4	1	1	1	0	0	0	2	0	1	1

i	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**	**64**
G_1	1	0	2	0	1	0	1	2	0	4	1	2	1	2	3	0	1	1	2	1	0	0	0	2	1	1	0	1	0	1	0	0
G_2	4	2	2	1	2	1	1	2	3	1	1	1	1	2	0	0	0	1	0	2	1	0	1	0	0	0	3	4	2	2	1	2
G_3	1	0	4	0	1	2	3	0	2	2	1	0	1	0	1	1	0	2	0	1	0	0	0	2	0	0	0	1	1	1	0	0

i	**65**	**66**	**67**	**68**	**69**	**70**	**71**	**72**	**73**	**74**	**75**	**76**	**77**	**78**	**79**	**80**	**81**	**82**	**83**	**84**	**85**	**86**	**87**	**88**	**89**	**90**	**91**	**92**	**93**	**94**	**95**	**96**
G_1	2	0	2	1	0	0	1	2	0	2	1	0	3	0	0	2	1	0	0	4	0	1	1	2	1	1	2	0	2	2	3	0
G_2	1	1	3	1	2	0	2	0	2	0	1	1	1	1	0	3	1	5	0	1	1	0	2	1	2	0	0	1	0	2	1	0
G_3	1	0	2	0	2	3	1	0	0	2	2	1	2	0	2	1	0	1	3	0	2	1	0	0	2	2	1	1	0	1	2	1

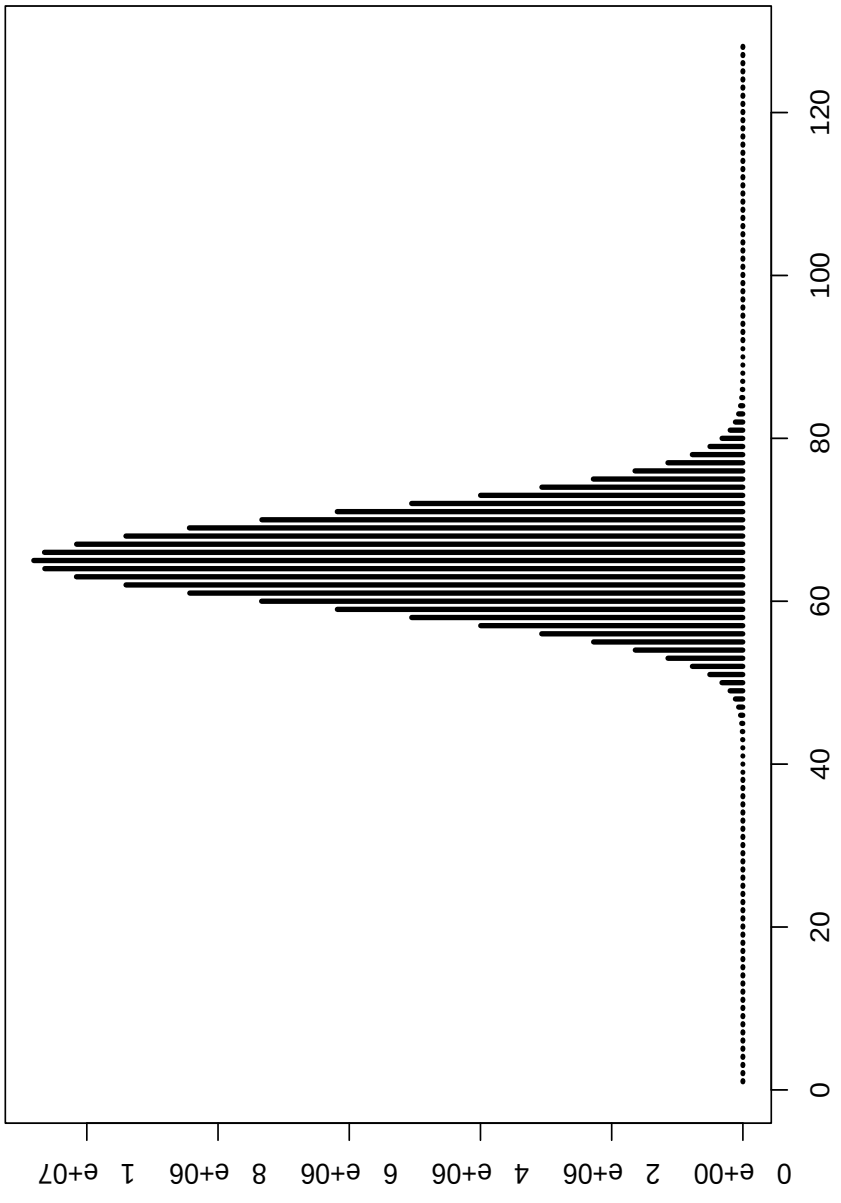
i	**97**	**98**	**99**	**100**	**101**	**102**	**103**	**104**	**105**	**106**	**107**	**108**	**109**	**110**	**111**	**112**	**113**	**114**	**115**	**116**	**117**	**118**	**119**	**120**	**121**	**122**	**123**	**124**	**125**	**126**	**127**	**128**
G_1	2	0	3	0	0	1	1	2	1	0	0	0	2	0	2	2	2	0	1	1	1	1	0	2	1	2	1	0	1	2	3	0
G_2	2	2	0	0	0	2	1	1	1	1	0	1	1	0	1	0	0	0	1	0	0	0	3	2	0	1	1	1	0	2	0	2
G_3	1	1	1	1	0	1	0	3	0	2	0	2	0	0	0	2	0	0	1	2	1	0	3	1	2	1	0	0	0	0	1	0

Tabulka A.2: Výsledky testu 2.2 na hladině $0,01$

A.2 Výsledky testu 2.4

i	G_1	G_2	G_3	$\sum G_r$
0–31	0	0	0	0
32	0	1	0	1
33	2	1	0	3
34	1	1	0	2
35	6	5	5	16
36	7	18	18	43
37	34	32	29	95
38	78	81	69	228
39	188	161	176	525
40	359	377	365	1101
41	842	826	827	2495
42	1686	1735	1743	5164
43	3336	3439	3446	10221
44	6534	6564	6611	19709
45	12273	12430	12363	37066
46	22153	22160	22270	66583
47	38742	38770	38651	116163
48	65472	65451	65567	196490
49	106899	106741	107081	320721
50	168893	168161	168525	505579
51	256840	257777	256826	771443
52	381193	380840	380898	1142931
53	547511	546915	547670	1642096
54	759160	758816	761148	2279124
55	1022375	1023003	1023117	3068495
56	1332174	1333517	1332033	3997724
57	1682944	1683476	1683476	5049896
58	2061007	2060860	2060021	6181888
59	2446161	2445150	2445603	7336914
60	2811234	2809879	2812599	8433712
61	3134866	3134718	3131393	9400977
62	3386623	3386609	3384303	10157535
63	3546264	3549298	3547783	10643345
MAX 64	3602793	3604556	3604202	10811551
65	3547836	3550916	3549329	10648081
66	3386889	3383703	3388688	10159280
67	3135811	3133708	3135370	9404889
68	2813392	2811129	2812890	8437411
69	2444174	2444934	2443732	7332840
70	2060802	2062703	2059926	6183431
71	1683486	1682792	1683571	5049849
72	1333397	1333199	1332397	3998993
73	1021494	1022440	1022829	3066763
74	760867	759125	758644	2278636
75	548533	548915	547798	1645246
76	381620	381188	381637	1144445
77	257124	258115	257444	772683
78	168007	167364	168523	503894
79	106623	105813	106066	318502
80	65054	65267	65542	195863
81	38638	38555	39210	116403
82	22153	22193	22156	66502
83	12270	12496	12298	37064
84	6590	6541	6521	19652
85	3404	3389	3372	10165
86	1689	1696	1641	5026
87	812	809	887	2508
88	395	364	415	1174
89	175	167	179	521
90	67	63	66	196
91	31	31	34	96
92	8	11	11	30
93	4	4	4	12
94	3	1	2	6
95	2	1	0	3
96–128	0	0	0	0

Tabulka A.3: Výsledky testu 2.4 s využitím všech tří generátorů a jejich součet



Obrázek A.1: Graf četnosti počtu změněných bitů z testu 2.4

A.3 Stevensovy podmínky

	[32] ...			... [1]	podmínky pro Z_i
Q_1, Q_2	`........`	`........`	`........`	`........`	
Q_3	`........`	`....0...`	`....0...`	`.0......`	
Q_4	`1.......`	`0^^^1^^^`	`^^^^0^^^`	`^011....`	
Q_5	`1000100.`	`01..0000`	`00000000`	`0010.1.1`	
Q_6	`0000001^`	`01111111`	`10111100`	`0100^0^1`	
Q_7	`00000011`	`11111110`	`11111000`	`00100000`	
Q_8	`00000001`	`1..10001`	`0.0.0101`	`01000000`	
Q_9	`11111011`	`...10000`	`0.1^1111`	`00111101`	
Q_{10}	`0111....`	`0..11111`	`1101...0`	`01....00`	
Q_{11}	`00100000`	`1...0001`	`11000000`	`11000010`	
Q_{12}	`000...00`	`....1000`	`0001...1`	`0.......`	
Q_{13}	`01....01`	`....1111`	`111....0`	`0...1...`	
Q_{14}	`0.0...00`	`....1011`	`111....1`	`1...1...`	
Q_{15}	`0.1...01`	`.......0`	`1.......`	`....0...`	
Q_{16}	`0!1.....`	`......!.`	`........`	`........`	
Q_{17}	`0!......`	`......0.`	`^.......`	`....^...`	
Q_{18}	`0.^.....` ($S1$)($S2$)	`......1.` ($S3$)	`........`	`........`	
Q_{19}	`0.......` ($S4$)	`......0.` ($S5$)	`........`	`........`	
Q_{20}	`0.......` ($S6$)	`.....!..` ($S7$)	`........`	`........`	
Q_{21}	`0.......` ($S8$)	`......^.` ($S9$)	`........`	`........`	
Q_{22}	`0.......` ($S10$)	`........`	`........`	`........`	
Q_{23}	`0.......` ($S12$)	`........`	`........`	`........`	$Z_{23}[18] = 0$ ($S11$)
Q_{24}	`1.......` ($S13$)	`........`	`........`	`........`	
Q_{25}–Q_{34}	`........`	`........`	`........`	`........`	
Q_{35}	`........`	`........`	`........`	`........`	$Z_{35}[16] = 0$ ($S14$)
Q_{36}–Q_{45}	`........`	`........`	`........`	`........`	

Tabulka A.4: Stevensovy podmínky pro zpracování prvního bloku (první část)

	[32] ...			... [1]	podmínky pro Z_i
Q_{46}	....	....	....	....	
Q_{47}	....	....	....	....	
Q_{48}	A....... ($S15$)	....	....	....	
Q_{49}	B....... ($S16$)	....	....	....	
Q_{50}	C....... ($S17$)	....	....	....	
Q_{51}	B....... ($S18$)	....	....	....	
Q_{52}	C....... ($S19$)	....	....	....	
Q_{53}	B....... ($S20$)	....	....	....	
Q_{54}	C....... ($S21$)	....	....	....	
Q_{55}	B....... ($S22$)	....	....	....	
Q_{56}	C....... ($S23$)	....	....	....	
Q_{57}	B....... ($S24$)	....	....	....	
Q_{58}	C....... ($S25$)	....	....	....	
Q_{59}	B....... ($S26$)	....	....	....	
Q_{60}	A....... ($S27$)	....	....	....	
Q_{61}	B....... ($S28$)	....	....	....	
Q_{62}	A....... ($S29$)	....	....	....	
Q_{63}	B....... ($S30$)	....	....	....	
Q_{64}	....	....	....	....	
I'_1	....	....	....	....	
I'_2	^....00. ($S31$)($S32$)($S33$)	....	....	..0..... ($S34$)	
I'_3	^....01. ($S35$)($S36$)($S37$)	....	....	....	
I'_4	......0. ($S38$)	....	....	....	

Vysvětlení:

`.` – libovolný bit, `0/1` – pevně daný bit,
`^` – bit stejný jako nad ním, `!` – opačný k bitu nad ním,
`A` – 32. bit Q_{46}, `B` – 32. bit Q_{47},
`C` – opačný k `A` (Si) – označení podmínky

Tabulka A.5: Stevensovy podmínky pro zpracování prvního bloku (druhá část)

A.4 Klímovy podmínky

	[32] ...			... [1]	podmínky pro Z_i
Q_1, Q_2	....	....	....	....	
Q_3	....	....0...	....0...	.0......	
Q_4	1.......	0^^^1^^^	^^^^0^^^	^011....	
Q_5	1000100.	01000000	00000000	0010.1.1	
Q_6	0000001^	01111111	10111100	0100^0^1	
Q_7	00000011	11111110	11111000	00100000	
Q_8	00000001	1..10001	0.0.0101	01000000	
Q_9	11111011	...10000	0.1^1111	00111101	
Q_{10}	0111....	00011111	1.01...0	01....00	
Q_{11}	0010.0.0	111.0001	1^00.0.0	11....10	
Q_{12}	000...^0	1000	0001...1	0.......	
Q_{13}	01....01	1111	111....0	0...1...	
Q_{14}	000...00	1011	111....1	1...1...	
Q_{15}	.1100001	....	10......	.0000000	
Q_{16}	^01000..	..!.....	....	.000.000	
Q_{17}	^1......	......0.	^.......	....^...	
Q_{18}	^.^..... $_{(K1)(k2)}$	......1. $_{(K3)}$	....	....	
Q_{19}	^....... $_{(K5)}$	......0. $_{(K6)}$	....	....	$Z_{19}[4,\ldots,18] \neq (1,\ldots,1)$ $_{(K4)}$
Q_{20}	^....... $_{(K8)}$	....	....	....	$Z_{20}[30,31,32] \neq (0,0,0)$ $_{(K7)}$
Q_{21}	^....... $_{(K9)}$	.......^. $_{(K10)}$	....	....	
Q_{22}	^....... $_{(K11)}$	....	....	....	
Q_{23}	0....... $_{(K13)}$	....	....	....	$Z_{23}[18] = 0$ $_{(K12)}$
Q_{24}	1....... $_{(K14)}$	....	....	....	
Q_{25}–Q_{34}	....	....	....	....	
Q_{35}	....	....	....	....	$Z_{35}[16] = 0$ $_{(K15)}$
Q_{36}–Q_{45}	....	....	....	....	

Tabulka A.6: Klímovy podmínky pro zpracování prvního bloku (první část)

	[32] ...			... [1]	podmínky pro Z_i
Q_{46}	.....	.....	.....	.....	
Q_{47}	.....	.....	.....	.....	
Q_{48}	A..... ($K16$)	.....	.....	.....	
Q_{49}	B..... ($K17$)	.....	.....	.....	
Q_{50}	C..... ($K18$)	.....	.....	.....	
Q_{51}	B..... ($K19$)	.....	.....	.....	
Q_{52}	C..... ($K20$)	.....	.....	.....	
Q_{53}	B..... ($K21$)	.....	.....	.....	
Q_{54}	C..... ($K22$)	.....	.....	.....	
Q_{55}	B..... ($K23$)	.....	.....	.....	
Q_{56}	C..... ($K24$)	.....	.....	.....	
Q_{57}	B..... ($K25$)	.....	.....	.....	
Q_{58}	C..... ($K26$)	.....	.....	.....	
Q_{59}	B..... ($K27$)	.....	.....	.....	
Q_{60}	A..... ($K28$)	.....	.....	.....	
Q_{61}	B..... ($K29$)	.....	.....	.....	
Q_{62}	A..... ($K30$)	.....	.....	.....	
Q_{63}	B..... ($K31$)	.....	.....	.....	
Q_{64}	.....	.....	.....	.....	
I'_1	.....	.....	.....	.....	
I'_2	^....00. ($K32$)($K33$)($K34$)	.....	.....	..0..... ($K35$)	
I'_3	^....01. ($K36$)($K37$)($K38$)	.....	.....	.....	
I'_4	......0. ($K39$)	.....	.....	.....	

Vysvětlení:

. – libovolný bit, 0/1 – pevně daný bit,
^ – bit stejný jako nad ním, ! – opačný k bitu nad ním,
A – 32. bit Q_{46}, B – 32. bit Q_{47},
C – opačný k A (Ki) – označení podmínky

Tabulka A.7: Klímovy podmínky pro zpracování prvního bloku (druhá část)

A.5 Výsledky testu 4.1

S	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
G_1	3	2	6	4	8	3	4	4	1	9	6	5	4	4	5	2	2	4	6	5
G_2	8	6	5	2	4	2	6	4	3	9	7	2	1	8	4	9	4	4	4	3
G_3	5	0	7	5	6	6	3	1	7	3	4	10	7	6	6	7	4	3	3	6

S	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**
G_1	6	6	2	2	6	4	3	5	3	3	8	5	4	5	10	3	3	7
G_2	6	5	2	5	4	6	3	4	9	6	9	4	6	8	6	2	6	8
G_3	2	6	2	4	2	8	8	6	4	1	11	5	6	4	6	5	5	5

Tabulka A.8: Výsledky testu 4.1 (Stevensovy pp) na hladině $0,05$

S	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
G_1	1	1	0	1	0	0	1	2	1	2	0	3	0	1	3	0	1	4	0	2
G_2	1	2	1	2	2	0	1	1	2	0	1	0	0	2	0	1	0	2	1	2
G_3	3	2	1	1	2	3	3	0	1	0	1	0	2	1	2	3	2	1	2	0

S	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**
G_1	0	0	1	0	3	1	3	0	1	0	2	1	0	0	0	0	1	1
G_2	1	0	0	2	0	1	2	1	0	2	1	0	1	0	0	4	0	2
G_3	1	0	1	1	1	3	0	1	0	0	2	0	2	2	1	0	0	3

Tabulka A.9: Výsledky testu 4.1 (Stevensovy pp) na hladině $0,01$

K	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
G_1	1	3	4	2	5	6	5	3	4	3	4	2	4	4	4	7	8	4	10	5
G_2	2	4	7	1	4	4	6	8	4	7	5	8	6	2	5	6	7	3	4	4
G_3	6	11	6	5	5	3	9	4	3	3	6	3	6	5	6	2	6	10	4	4

K	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
G_1	7	3	9	3	2	4	2	4	5	4	3	5	10	3	1	5	6	4	6
G_2	5	7	3	8	3	3	4	1	3	3	5	13	8	4	8	4	10	7	7
G_3	6	10	6	4	3	8	4	2	6	3	5	7	5	5	7	5	1	6	4

Tabulka A.10: Výsledky testu 4.1 (Klímovy pp) na hladině $0,05$

K	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
G_1	1	1	1	4	0	2	0	3	2	1	2	0	0	1	0	0	1	0	1	3
G_2	1	3	3	2	3	0	0	0	1	1	1	0	4	0	1	0	0	2	1	0
G_3	2	4	5	3	3	0	2	0	2	0	3	0	4	1	0	0	2	0	2	0

K	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
G_1	2	0	0	2	1	2	0	0	2	2	0	2	0	1	1	1	0	2	0
G_2	2	0	1	1	0	2	3	1	1	0	1	0	1	0	0	0	1	3	0
G_3	0	0	1	0	0	1	0	2	0	0	1	2	1	0	1	1	0	3	0

Tabulka A.11: Výsledky testu 4.1 (Klímovy pp) na hladině $0,01$

A.6 Výsledky testu 4.2

S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
1	7	7	100	8	8	6	3	2	5	5	4	6	9	2	2	5	4	5	3	4	2	9	6	4	7	5	3	4	9	3	5	5	5	7	12	6	5	
2		6	9	6	3	3	5	1	4	8	5	6	3	5	7	7	5	3	7	5	4	2	6	3	6	6	4	9	5	4	3	4	1	4	2	4	2	
3			5	13	3	5	2	4	0	2	8	4	4	6	7	4	5	2	4	5	5	5	7	6	6	5	8	3	5	2	1	6	4	7	5	6	8	
4				5	2	2	2	1	6	7	4	5	4	4	0	2	2	2	9	6	6	3	7	7	3	5	4	5	3	4	5	4	5	7	5	7	6	
5					6	2	4	2	4	5	3	6	3	7	5	5	4	6	1	4	3	5	7	8	2	5	4	2	4	7	4	5	4	3	3	10	6	
6						2	5	2	0	6	5	3	4	5	3	4	8	6	11	6	7	8	7	9	0	3	7	6	1	6	2	5	4	6	3	6	6	
7							10	7	4	7	8	3	8	8	3	3	8	3	5	5	6	3	7	6	5	6	4	6	7	4	3	8	10	6	8	5	8	
8								1	4	3	6	3	3	4	1	3	5	4	6	4	5	4	3	3	2	4	5	8	4	5	3	4	3	2	2	4	6	
9									6	7	2	7	4	2	4	3	7	8	4	9	6	5	6	2	4	9	3	6	3	5	3	4	5	4	2	4	6	
10										7	5	8	10	7	8	7	4	4	8	2	2	4	7	5	2	8	2	5	3	5	5	4	10	8	7	5	4	
11											4	4	4	2	7	3	10	3	5	5	2	2	3	1	3	5	4	7	5	3	8	6	6	3	4	6	7	
12												5	3	4	5	8	4	9	6	5	8	4	4	3	8	7	5	9	4	9	3	4	8	3	6	5	6	
13													2	3	5	5	4	10	10	3	4	6	4	2	4	5	4	11	5	6	1	8	7	4	4	9	6	
14														4	3	9	4	7	5	7	3	4	1	13	4	2	8	5	2	3	5	7	2	4	8	7	5	
15															3	6	8	5	4	9	1	3	5	4	5	1	5	8	5	9	9	5	1	6	5	12	0	
16																4	4	4	9	4	4	6	11	7	6	5	7	5	8	6	6	4	4	3	4	4	3	
17																	5	5	5	2	6	6	4	5	2	6	7	6	5	6	4	4	1	7	1	5	5	
18																		2	2	5	3	3	9	8	6	10	7	6	3	2	5	5	4	4	6	7	6	
19																			5	3	4	4	5	6	5	4	11	11	5	4	5	4	4	5	3	4	5	
20																				3	5	6	7	2	5	8	4	8	4	5	4	5	3	11	7	3	6	
21																					6	6	5	9	2	5	4	5	4	3	4	6	1	3	4	6	2	
22																						3	6	5	1	4	6	6	2	7	7	5	7	6	3	7	7	
23																							5	4	2	8	12	3	3	4	3	6	4	5	9	10	7	
24																								3	7	7	7	2	3	7	5	2	4	5	2	3	5	
25																									5	4	5	5	1	5	7	7	6	3	2	6	3	
26																										6	8	6	5	4	7	1	6	5	5	8	7	
27																											6	6	4	5	4	3	7	5	6	6	2	
28																												9	5	3	12	6	4	4	3	10	4	
29																													4	7	2	3	7	4	7	9	3	
30																														6	5	1	5	1	8	10	5	
31																															4	5	3	6	5	6	3	
32																																7	8	2	2	6	8	
33																																	2	6	5	3	6	
34																																		7	4	5	5	
35																																			5	6	8	
36																																				4	4	
37																																					4	
38																																						

Tabulka A.12: Výsledky testu 4.2 (Stevensovy pp) na hladině $0,05$ pro generátor G_2

S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
1	0	0	1	0	0	1	1	0	2	0	0	1	2	1	2	1	0	0	2	0	1	0	2	0	1	0	0	2	0	1	0	1	0	1	0	1	0	1	0
2			1		2		1	0	0	4	1	2	0	0	1	0	0	3	0	0	2	1	1	0	1	0	2	0	1	0	0	2	1	5	1	1	1	1	1	1
3					3		5	2	1	1	2	1	1	2	1	1	1	0	0	3	1	0	1	2	0	0	1	2	0	0	0	3	1	0	1	0	2	0	1	0
4							0	3	0	1	0	1	5	0	1	1	0	0	1	2	3	1	2	0	2	0	3	1	1	2	0	0	2	0	1	1	0	1	3	2
5								3	0	3	0	1	2	0	1	0	3	0	0	0	2	3	1	1	0	0	1	0	0	0	2	0	1	0	0	0	1	0	2	3
6									1	0	0	1	0	1	1	0	0	0	2	1	2	0	1	1	3	3	0	2	1	3	1	0	1	1	1	2	1	1	0	0
7										1	1	2	0	0	3	1	1	1	0	1	0	0	4	1	3	0	2	2	5	0	1	0	1	0	0	0	2	0	2	1
8											2	2	3	3	2	1	0	2	1	2	0	1	0	0	1	1	1	0	0	0	1	1	1	1	0	1	0	1	1	1
9												0	0	1	2	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	1	0	6	1	0	2	0	0	1	1
10													2	1	1	2	0	1	1	1	0	2	1	3	0	1	0	1	1	0	0	1	1	1	0	4	2	0	0	2
11														3	2	3	0	2	1	0	0	0	3	0	1	0	1	1	0	3	1	1	0	0	4	0	3	0	0	2
12															2	4	0	0	2	1	0	4	1	2	0	1	1	1	1	3	0	1	2	1	3	0	1	1	1	1
13																0	1	0	0	1	0	1	3	0	1	1	1	0	0	1	1	0	0	1	3	0	1	0	0	1
14																	1	1	0	0	1	2	0	0	0	2	2	0	0	1	1	1	1	1	1	1	0	1	1	1
15																		2	2	2	2	1	2	1	1	2	0	0	1	1	2	3	1	0	2	0	0	1	1	0
16																			0	0	2	0	0	0	0	1	0	2	3	1	2	2	0	1	0	1	2	4	0	0
17																				0	2	1	1	1	0	1	1	0	2	2	1	2	1	1	0	0	0	1	0	2
18																					1	1	1	0	2	0	0	0	2	0	1	1	0	0	5	0	0	3	0	0
19																						3	1	2	3	0	0	2	2	2	1	1	1	0	2	1	0	2	2	1
20																							3	1	1	2	0	3	0	2	2	2	1	1	0	1	0	3	0	2
21																								1	0	0	0	0	0	0	2	0	1	7	1	1	0	1	2	2	0
22																									0	1	1	1	0	1	1	2	1	0	2	0	2	0	3	2
23																										0	2	2	1	0	1	1	0	1	0	1	1	0	0	2
24																											1	0	1	1	1	0	0	0	2	0	2	2	1	0
25																												3	0	1	2	0	0	1	1	4	0	3	3	1
26																													2	1	1	0	0	1	1	1	0	2	0	1
27																														0	1	3	1	4	0	0	0	0	0	2
28																															2	0	0	1	0	3	2	2	3	1
29																																0	1	0	2	0	3	3	1	2
30																																	1	1	2	3	0	0	1	0
31																																		1	1	1	1	0	0	0
32																																			0	2	3	0	2	2
33																																				2	0	1	1	1
34																																					0	0	1	0
35																																						2	0	1
36																																							0	0
37																																								0
38																																								

Tabulka A.13: Výsledky testu 4.2 (Stevensovy pp) na hladině $0,01$ pro generátor G_2

K	1	2	3	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
1	7	5	4	2	7	10	8	7	3	7	8	5	6	6	3	2	2	5	1	6	3	7	9	1	6	3	4	4	6	9	4	4	6	5	2	6	5
2	3	2	8	3	6	8	2	3	5	0	5	6	3	5	3	6	6	6	6	6	5	4	6	8	5	7	5	2	3	4	4	6	5	3	4	6	6
3	2	100	2	4	5	11	4	2	7	1	4	0	5	6	7	4	6	7	7	3	6	4	4	10	8	1	5	5	6	5	4	3	5	8	6
5	7	4	6	6	1	6	8	6	5	5	6	3	2	9	6	6	10	10	2	5	2	7	6	4	2	3	1	3	5	4	6	3	5	6
6	3	6	4	4	6	5	4	7	4	3	5	8	5	2	4	1	4	3	4	4	5	4	4	9	6	4	0	2	2	2	7	5	2
7	6	2	3	4	8	10	4	1	3	2	4	4	7	6	4	7	6	3	6	2	12	8	1	3	2	6	10	3	6	6	4	5	7	4	4
8	3	7	4	6	4	5	7	6	8	3	11	6	2	2	6	3	2	8	3	8	7	6	7	5	3	1	4	5	5	9	11
9	7	10	6	3	4	6	4	9	14	5	4	12	3	8	4	4	4	5	4	6	4	3	3	2	6	3	5	6	5	7
10	6	6	0	5	6	7	3	8	4	6	6	5	7	3	6	4	3	5	6	6	5	7	7	3	4	7	8	2	6	6	4	3	5	6	3	6	5	6
11	5	7	5	5	8	2	5	9	6	5	3	3	4	7	8	8	8	5	3	7	5	4	6	2	7	7	1	5
12	5	6	3	2	9	3	5	8	1	7	7	6	3	4	3	4	6	4	3	11	5	9	0	2	5	5	1
13	5	5	4	7	6	8	0	8	2	3	5	7	5	5	3	7	5	3	7	4	4	1	2	7	4	6
14	8	4	8	1	2	3	4	5	3	8	4	10	6	7	6	6	3	4	4	2	3	6	5	4	7
15	7	8	4	5	3	3	6	4	5	7	5	6	6	3	1	4	5	7	5	7	4	7	2	4
16	5	5	2	8	12	2	6	1	2	5	5	5	6	4	2	6	3	3	4	5	6	2	3
17	4	10	0	9	7	5	1	2	4	2	4	5	5	5	4	6	3	10	5	3	4	1
18	1	4	7	5	1	4	7	6	10	2	4	4	6	6	2	8	4	5	6	9	5
19	3	5	4	3	4	2	9	4	7	6	6	5	3	6	5	3	3	5	5	1	1
20	1	5	2	3	4	6	3	7	8	5	4	4	9	3	8	2	5	7	5
21	5	9	3	3	9	4	4	3	4	1	8	5	6	4	5	9	7	4
22	6	4	8	6	3	4	3	10	1	3	5	3	8	4	6	4	7
23	2	5	7	3	9	2	4	4	2	4	6	5	9	2	9	4
24	2	7	4	5	10	8	7	1	0	3	5	4	5	5	5
25	3	3	7	3	5	6	4	3	3	2	2	11	2	3
26	6	3	5	10	10	5	4	7	6	6	5	2	3
27	4	2	3	5	9	4	5	6	5	4	0	6
28	2	3	8	5	8	3	4	2	4	7	7
29	4	3	2	4	3	4	6	7	6	6
30	4	2	3	6	8	9	3	5	7
31	1	3	1	4	6	10	6	5
32	3	5	6	5	6	1	4
33	4	3	7	4	3	3
34	11	7	7	5	2
35	5	3	4	5
36	6	2	8
37	0	2
38	4
39	

Tabulka A.14: Výsledky testu 5 (Klímovy pp) na hladině $0,05$ pro generátor G_2

K	1	2	3	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
1	0	1	2	2	0	0	2	0	0	1	2	4	0	2	2	2	0	3	0	1	0	1	0	3	1	1	1	1	0	0	1	0	2	4	0	0	1	
2	2	1	0	1	1	3	0	2	3	1	3	1	1	1	1	1	0	1	1	2	3	4	2	2	2	1	0	2	1	2	3	2	2	2	0	1	1	
3	3	100	1	1	0	1	1	1	1	0	2	0	0	1	1	1	1	0	1	0	0	0	0	0	1	1	3	2	2	1	1	2	0	1	1	1		
5	0	0	1	0	0	2	2	3	1	2	1	0	0	1	1	2	0	1	2	1	1	1	1	3	1	0	2	3	0	1	2	0	1	2				
6	0	1	0	0	0	2	1	2	0	0	1	1	0	1	1	1	2	0	0	0	2	0	1	2	1	0	0	1	0	1	1	0	1					
7	2	0	1	0	1	3	0	1	1	0	0	2	1	0	0	0	1	0	1	1	0	2	3	2	0	1	2	1	0	2	1	3						
8	2	0	2	0	2	0	2	2	1	2	2	1	4	2	1	3	2	1	0	0	0	0	3	1	1	1	1	1	1	0	0							
9	1	2	4	0	0	1	0	1	0	0	2	1	0	2	1	2	1	1	0	0	2	3	2	1	1	2	0	1	3	3								
10	0	2	2	1	1	3	0	3	1	1	2	1	0	0	1	2	2	2	2	2	0	2	1	1	0	1	0	2	2	0								
11	1	0	0	1	0	3	1	1	0	1	2	0	1	1	0	0	0	1	2	2	0	1	1	2	0	1	1	2										
12	1	0	2	2	1	2	2	2	0	1	0	2	0	0	1	1	1	0	0	1	1	3	0	0	2	2	3											
13	1	3	0	1	0	3	1	2	2	1	0	0	0	2	2	4	1	1	1	2	0	1	0	1	0	1												
14	0	2	3	1	0	0	2	0	0	1	1	1	1	1	1	2	3	0	0	0	1	1	1	1	0													
15	1	1	1	1	3	2	1	2	1	0	2	2	0	1	1	1	2	1	0	1	0	1	1	1														
16	0	0	1	1	0	2	0	2	0	1	2	1	1	0	0	1	2	0	1	2	2	2	0															
17	0	2	0	0	1	0	0	1	0	1	0	2	4	1	2	1	1	0	1	1	1	1																
18	0	1	2	0	1	1	2	1	0	1	1	1	0	1	3	1	0	2	0	0	2																	
19	1	2	0	1	0	0	0	2	1	0	0	1	3	3	0	1	1	0	1	4																		
20	2	0	1	1	1	0	0	1	0	3	2	2	0	2	0	1	1	0	1																			
21	0	3	1	3	2	1	1	1	0	1	0	2	0	1	0	0	3	0																				
22	3	1	0	0	0	1	1	1	1	0	0	2	0	1	3	2	0																					
23	2	0	2	1	2	0	2	0	0	1	1	2	1	1	2	0																						
24	0	2	2	3	0	1	0	0	1	0	1	2	0	1	0																							
25	0	0	1	0	1	1	2	1	0	3	0	0	0	0																								
26	2	0	1	4	0	1	1	4	1	0	0	0	1																									
27	0	2	0	1	2	0	1	1	2	2	1	2																										
28	0	1	3	0	0	1	0	1	0	1	0																											
29	4	0	2	3	1	1	1	1	0	0																												
30	2	0	1	2	2	0	0	0	0																													
31	0	0	0	1	1	4	0	0																														
32	2	3	3	3	0	0	1																															
33	1	2	1	1	1	0																																
34	2	0	0	0	2																																	
35	0	1	1	1																																		
36	1	3	2																																			
37	2	3																																				
38	2																																					
39																																						

Tabulka A.15: Výsledky testu 5 (Klímovy pp) na hladině $0,01$ pro generátor G_2

Dodatek B

Zdrojový kód

Zdrojový kód je spolu s výsledky, které se nevešly do tištěné podoby, na přiloženém CD.