Slovenská Technická Univerzita v Bratislave
Fakulta Elektrotechniky a Informatiky

Moderná kryptoanalýza klasických šifier

Dizertačná práca

Bratislava 2017 **Ing. Eugen Antal**

Slovenská Technická Univerzita v Bratislave
Fakulta Elektrotechniky a Informatiky
Ústav Informatiky a Matematiky

Ing. Eugen Antal

Dizertačná práca

Téma dizertačnej práce:	Moderná kryptoanalýza klasických šifier
Evidenčné číslo:	FEI-104372-36087
Školiteľ:	prof. RNDr. Otokar Grošek, PhD.
Konzultant:	Mgr. Marek Sýs, PhD.
Forma štúdia:	denná
Začiatok štúdia:	1.9.2011
Študijný program:	Aplikovaná informatika
Študijný odbor:	9.2.9 Aplikovaná informatika

2017

Čestné vyhlásenie

Čestne vyhlasujem, že som túto písomnú prácu vypracoval sám s použitím uvedenej literatúry. V prípade vlastných publikácií je spoluautorstvo explicitne uvedené.

V Bratislave, 07.02.2017

.....

Ing. Eugen Antal

Chcem sa poďakovať prof. RNDr. Otokarovi Grošekovi, PhD. a Mgr. Marekovi Sýsovi, PhD. za cenné rady, pripomienky a obetovaný čas pri vedení tejto práce a počas môjho doterajšieho pôsobenia na Ústave informatiky a matematiky.
Moja vďaka patrí aj Mgr. Jakubovi Mírkovi za poskytnutú pomoc pri získavaní šifrovaných listov z českých archívov ako aj doc. Ing. Pavlovi Zajacovi, PhD. a RNDr. Karle Čipkovej, PhD.
Ďalej by som sa chcel poďakovať svojej priateľke Mgr. Annamárii Oláhovej a rodine za poskytnutú pomoc a morálnu podporu.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam Skratiek a značiek

AJ	Anglický jazyk.
ACO	Optimalizácia pomocou kolónie mravcov (Ant Colony Optimisation).
$\mathbb{B}$	Počet lepšie ohodnotených susedov.
$\mathbb{F}$	Množina účelových funkcií.
$\mathcal{F}_{max}$	Globálny extrém účelovej funkcie.
FFA	Firefly algorithm.
GA	Genetický algoritmus (Genetic Algorithm).
HC	Horolezecký algoritmus (Hill Climbing).
$hMax$	Homogénna časť s najväčším počtom rovnako ohodnotených extrémov.
IC	Index koincidencie.
JSD	Jensen-Shannonova divergencia.
KL	Kullback-Leibler divergencia.
$\mathbb{M}$	Percentuálna zhoda grafém textu so správnym výsledkom.
M_n	Markovský model rádu $n-1$, založený na frekvencii n-gramoch.
MR	Meranie vyhladenosti ("measure of roughness").
OANC	Open American National Corpus.
OT	Otvorený text.
PSO	Particle swarm optimisation.
DPSO	Diskrétna verzia PSO.
RS	Náhodné prehľadávanie (Random Search).
SA	Simulované žíhanie (Simulated Annealing).
SAHC	Steepest Ascent Hill Climbing.
SI	Swarm Intelligence.
TS	Tabu search.
TSA	Telegrafná (anglická) abeceda, má 26 znakov.
TSP	Problém obchodného cestujúceho (Travelling Salesman Problem).
ZT	Zašifrovaný (alebo zatvorený) text.
Z_n	Množina celých čísiel, t.j. $\{0, 1, \ldots, n-1\}$.
θ_{corr}	Správne riešenie.

Úvod

Predkladaná práca je zhrnutím aktuálnych poznatkov modernej kryptoanalýzy klasických šifier a zameriava sa na možnosť využitia moderných optimalizačných metód pri ich lúštení.

Klasické šifry predstavujú pomerne dobre definovanú oblasť kryptológie. Rýchly vývin výpočtových technológií umožnil modernú a rýchlu kryptoanalýzu týchto šifier. Napriek tomu, že väčšina klasických šifier neodolá moderným technikám počítačového lúštenia, existujú isté triedy klasických šifier, ktoré úspešne odolávajú aj najmodernejším postupom. Príkladom takýchto šifier sú homofónne substitučné šifry ktoré, v prípade dobrej konštrukcie, odolávajú doposiaľ známym metódam počítačového lúštenia.

Optimalizačné algoritmy - nazývané aj ako meta-heuristiky - patria do skupiny moderných spôsobov riešenia zložitých problémov. V zjednodušenom chápaní sa jedná o prehľadávanie možných parametrov tzv. účelovej funkcie, ktoré predstavujú možné riešenia stanoveného problému. Cieľom prehľadávania je nájdenie tých parametrov funkcie, ktoré reprezentujú najlepšie riešenie - globálny extrém účelovej funkcie. Pod pojmom optimalizácia chápeme hľadanie extrému účelovej funkcie.
Napriek tomu, že tieto metódy sa používajú väčšinou len na aproximáciu správneho riešenia (čiže nie na nájdenie najlepšieho konkrétneho riešenia, ale na nájdenie dostatočne dobrého), našli si svoje uplatnenie v širokom spektre rôznych tried ťažkých (aj NP-ťažkých) problémov. Vo väčšine prípadov sú tieto algoritmy založené na inteligentnom prehľadávaní veľkého priestoru riešení, alebo na aplikovaní rôznych prírodou inšpirovaných techník.

Jedným spôsobom kryptoanalýzy klasických šifier je využitie moderných optimalizačných metód. Principiálne sa jedná o transformáciu kryptoanalýzy na optimalizačný problém. Účelová funkcia v prípade lúštenia klasických šifier musí byť skonštruovaná tak, aby vyjadrovala mieru zmysluplnosti textu. Túto problematiku môžeme zaradiť do oblasti analýzy textu.
Napriek tomu, že v súčastnosti sa považuje táto metodika za pomerne dobre preskúmanú, klasické šifry patria do triedy problémov, kde ostáva otvorených veľa otázok. Takéto otázky sú napr. ako správne skonštruovať účelovú funkciu, alebo ako správne transformovať lúštenie šifier na optimalizačný problém. V doterajších prácach absentuje aj hlbšia analýza kľúčových častí lúštenia ako aj popis ucelenej metodiky. V prak-

tickej časti našej práce predkladáme preto nový ucelený prístup, ktorý by prispel k rozšíreniu súčasných možností počítačového lúštenia klasických šifier.
Výzvou doposiaľ zostávajú tie typy klasických šifier, v prípade ktorých lúštenie pomocou meta-heuristík nedáva dostatočne dobré výsledky. Jedným z nich je homofónna substitúcia, na ktorú v tomto kontexte nazeráme ako na vylepšenú verziu monoalfabetickej. Ako cieľ sme si stanovili vytvorenie metodiky na efektívne lúštenie monoalfabetickej substitúcie, ktorú následne rozšírime a aplikujeme na lúštenie homofónnej substitúcie.

Práca je členená na tri logické časti. Prvá časť (kapitoly 1, 2 a 3) je venovaná popisu súčastného stavu problematiky klasických šifier, analýzy textu a meta-heuristík. Z kritického zhodnotenia doterajších výsledkov a súčastného stavu vyvstávajú ciele tejto práce, ktoré sú vypracované a zhodnotené v kapitolách 4 a 5. Vo všetkých vykonaných experimentoch ako aj pri spracovaní výsledkov sme použili vlastnú implementáciu, naprogramovanú v programovacom jazyku Java. Na overenie kvality vytvorenej metodiky uvádzame v kapitole 6 výsledky lúštenia niekoľkých vybraných substitučných šifier, ktoré sa zachovali v rôznych archívoch a publikáciách.

Kapitola 1

Klasické šifry

V tejto kapitole zadefinujeme základné pojmy a základnú kategorizáciu klasických šifier. Uvedieme spôsoby automatického a poloautomatického počítačového lúštenia týchto šifier. Poukážeme pritom aj na dôležitosť analýzy textu z hľadiska úspešného počítačového lúštenia klasických šifier.

1.1 Základné pojmy

Komunikácia, čiže rôzne podoby výmeny a zdieľania informácií tvoria základ každej ľudskej spoločnosti. Myšlienka utajenia informácií a pokus o získanie takýchto informácií siahajú až do obdobia samotného začiatku ľudskej komunikácie. Preto sa na ochranu informácií začali vyvíjať rôzne postupy. Tieto postupy spolu s metódami na narušenie tejto ochrany a ich štúdium sa nazýva kryptológia [29]. Kryptológia sa delí na kryptografiu (metódy na návrh systémov utajenia) a kryptoanalýzu (metódy na narušenie systémov utajenia) [29].
Jedným spôsobom utajenia informácií je prevedenie informácie (napr. správy) z čitateľnej podoby, ktorá sa nazýva otvorený text (OT) na nečitateľnú podobu nazývanú zatvorený text (ZT). Túto metodiku všeobecne nazývame **šifrovanie**. Samozrejme je nutné aby existovala metóda na spätné prevedenie informácie z nečitateľnej podoby do pôvodnej podoby, ktorú nazývame **dešifrovanie**. Systém, resp. algoritmy na šifrovanie a dešifrovanie, nazývame **kryptosystém**. [30, 29]

Na presné vyjadrenie kryptosystému môžeme použiť Definíciu 1, ktorá pokrýva všeobecnú definíciu takýchto systémov.

Definícia 1 *Kryptosystém je pätica* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$*, ktorá spĺňa nasledovné podmienky [32]:*

- $\mathcal{P}$ *je množina otvorených textov zapísaných pomocou znakov abecedy otvorených textov* $\mathcal{A}_P$ *(*$\mathcal{P} \subset \mathcal{A}_P^*$*);*

- $\mathcal{C}$ *je množina zašifrovaných textov zapísaných pomocou znakov abecedy zašifrovaných textov* $\mathcal{A}_C$ *(*$\mathcal{C} \subset \mathcal{A}_C^*$*);*

- *$\mathcal{K}$ je množina všetkých možných kľúčov;*
- *$\mathcal{E}$ je množina šifrovacích transformácií e_k ($e_k : \mathcal{P} \to \mathcal{C}$) parametrizovaných kľúčom $k \in \mathcal{K}$;*
- *$\mathcal{D}$ je množina dešifrovacích transformácií d_k ($d_k : \mathcal{C} \to \mathcal{P}$) parametrizovaných kľúčom $k \in \mathcal{K}$;*
- *Šifrovací algoritmus priraďuje ľubovolnému konečnému reťazcu $X = x_0x_1 \dots x_n$, $X \in \mathcal{P}$ zašifrovaný reťazec $Y = y_0y_1 \dots y_n$, $Y \in \mathcal{C}$, kde platí $d_k(e_k(X)) = X$.*

Klasické šifry (nazývané aj ručné šifry) tvoria špeciálnu skupinu šifrovacích algoritmov (kryptosystémov). Základnou charakteristikou týchto šifier je ich použiteľnosť v tzv. "poľných podmienkach". To znamená, že šifrovanie a dešifrovanie musia byť uskutočniteľné jednoducho - pomocou pera a papiera alebo použitím jednoduchých (elektro) mechanických pomôcok. Do skupiny klasických šifier sa zaraďujú šifry vytvorené do konca druhej svetovej vojny, kde hrali dôležitú rolu [30, 29].
Nástupom moderných počítačov významnosť klasických šifier poklesla. Väčšina týchto šifier v dnešnej dobe totiž neodolá dostupnej výpočtovej sile a je v súčasnosti pokladaná za prekonanú a málo bezpečnú na používanie. Napriek tomu, existujú triedy šifier, ktoré predstavujú výzvu aj pre moderné techniky kryptoanalýzy [29].
Špeciálnu skupinu kryptosystémov tvoria zložitejšie šifrovacie stroje, tzv. rotorové šifrátory. Tieto stroje sa používali do nástupu moderných počítačov a fungovali na mechanickom, alebo elektro-mechanickom princípe[1] [30, 43]. Rotorové šifry sa považujú za medzistupeň medzi klasickými a modernými šiframi. Na základe mechanizmu svojho fungovania by sa však mohli zaradiť do skupiny klasických šifier.

Vo všeobecnosti v prípade klasických šifier je ako množina $\mathcal{A}_P$ resp. $\mathcal{A}_C$ použitá telegrafná abeceda (znaky $a, b, c, \dots, z$), alebo číselná reprezentácia použitých znakov - Z_n. Šifrovanie a dešifrovanie je parametrizované na základe kľúča $k \in \mathcal{K}$, ktorý sa vo väčšine prípadov klasických šifier zapisuje pomocou tabuľkovej formy, kde sa uvádza spôsob zmeny znakov z $\mathcal{A}_P$ na znaky z $\mathcal{A}_C$ a opačne. [29]

Vlastnosti klasických šifier sa od vlastností dnešných šifrovacích algoritmov odlišujú v niekoľkých zásadných bodoch [29]:

- V prípade väčšiny klasických šifier je veľkosť množín $\mathcal{P}, \mathcal{C}, \mathcal{K}$ menšia ako v prípade dnešných šifier.
- V prípade klasických šifier je väčšinou utajený celý algoritmus (nie len kľúč) a znalosť algoritmu môže viesť priamo k rozlúšteniu správy.
- V prípade klasických šifier - na rozdiel od dnešných - je možné využiť štatistické vlastnosti jazyka pri kryptoanalýze.

[1] Najznámejším príkladom rotorového šifrátora je šifrovací stroj Enigma [87].

1.2 Klasifikácia

Klasické šifry môžeme kategorizovať podľa rôznych kritérií, ako napr. použité operácie pri šifrovaní, alebo vzťah jednotlivých znakov abecedy otvoreného textu ($\mathcal{A}_P$) k znakom abecedy zašifrovaného textu ($\mathcal{A}_C$). Na základe vzťahu znakov OT k znakom ZT môžeme klasické šifry rozdeliť na transpozičné, substitučné a ich kombináciu.

Transpozičné šifry sú šifry, ktoré aplikujú pevne zvolenú permutáciu na bloky otvoreného textu. Transpozíciu môžeme zadefinovať nasledovným spôsobom [29]:

Definícia 2 *Nech $\mathcal{A}^n$ je množina reťazcov dĺžky n nad abecedou $\mathcal{A}$. Nech $x = x_1x_2\ldots x_n \in \mathcal{A}^n$ je ľubovolný reťazec dĺžky n. Nech π je permutácia na množine $I_n = \{1, 2, \ldots, n\}$, t.j. $\pi \in S_n$ (množina všetkých permutácií na I_n).*
Potom transpozícia reťazca x bude reťazec $y \in \mathcal{A}^n$:

$$y = \pi(x) = x_{\pi(1)}x_{\pi(2)}\ldots x_{\pi(n)}.$$

Z definície vyplýva, že v prípade transpozičných šifier ZT zachováva frekvenciu znakov (grafém) OT, čo predstavuje možnú slabinu, ktorá sa dá využiť pri lúštení. Šifrovanie je v tomto prípade len preusporiadané poradie znakov otvoreného textu [29].

Substitučná šifra je aplikácia substitúcie (nahradenie) znakov otvoreného textu na znaky zašifrovaného textu podľa vopred stanovených pravidiel. Šifrovanie a dešifrovanie prebieha znak po znaku, kde $\pi(x_i)$ je transformácia z $\mathcal{A}_P$ na $\mathcal{A}_C$ a $\pi^{-1}(y_i)$ je transformácia z $\mathcal{A}_C$ na A_P [29]:

$$Y = y_1y_2\ldots y_n = e_k(x_1x_2\ldots x_n) = \pi(x_1)\pi(x_2)\ldots\pi(x_n)$$
$$X = x_1x_2\ldots x_n = d_k(y_1y_2\ldots y_n) = \pi^{-1}(y_1)\pi^{-1}(y_2)\ldots\pi^{-1}(y_n)$$

Substitučné šifry sa rozdeľujú na monoalfabetické, polyalfabetické a homofónne. Tieto podkategórie sú charakterizované podľa vzťahu prvkov z množiny $\mathcal{A}_P$ k prvkom z množiny $\mathcal{A}_C$.
V prípade *monoalfabetickej* substitúcie šifrovací algoritmus e_k je bijektívne zobrazenie $\mathcal{A}_P$ na $\mathcal{A}_C$.
V prípade *polyalfabetickej* substitúcie sa jedná o aplikovanie viacero substitúcií $(\pi_1, \ldots, \pi_n)$. Ten istý znak z $\mathcal{A}_P$ môže byť zašifrovaný na rôzne znaky z $\mathcal{A}_C$. Šifrovanie a dešifrovanie sa nedajú popísať zobrazením prvkov množín $\mathcal{A}_P$ alebo $\mathcal{A}_C$.
Homofónna substitúcia je taká substitučná šifra, kde jeden znak z $\mathcal{A}_P$ sa môže zašifrovať na niekoľko znakov z $\mathcal{A}_C$. Dešifrovací algoritmus d_k je surjektívne zobrazenie prvku z $\mathcal{A}_C$ na prvok z $\mathcal{A}_P$. Pre homofónnu substitúciu platí, že $|\mathcal{A}_P| < |\mathcal{A}_C|$.

Rozdiel medzi vymenovanými substitučnými šiframi môžeme ozrejmiť nasledujúcim príkladom:

[2]Poradie v lexikografickom usporiadaní znakov abecedy, začínajúc prvým písmenom 'a' ako 0.

OT:	s	p	r	a	v	a
OT (číselná reprezentácia[2])	18	15	17	0	21	0
ZT (monoalfabetická)	1	2	3	4	5	4
ZT (homofónna)	1	2	3	4	5	6
ZT (polyalfabetická)	2	4	4	6	6	7

Tabuľka 1.1: Príklad substitučných šifier

1.3 Kryptoanalýza klasických šifier

Kryptoanalýzu klasických šifier môžeme vo veľmi jednoduchom chápaní považovať za snahu o preloženie zašifrovaného textu na človekom čitateľný zmysluplný text. Pod pojmom kryptoanalýza teda chápeme metódy, ktorých cieľom je správne dešifrovanie ZT, resp. nájdenie správneho kľúča [32].

Pri lúštení klasických šifier sa väčšinou vyskytujú nasledujúce dve [32] situácie:

- Nepoznáme kategóriu šifry, k dispozícii máme len zašifrovaný text.
- Poznáme kategóriu šifry (dešifrovací algoritmus $d_k \in \mathcal{D}$) a zašifrovaný text.

V nasledujúcej časti podrobnejšie rozoberieme jednotlivé kroky kryptoanalýzy.

1.3.1 Určenie kategórie šifry

Pri lúštení klasických šifier vo väčšine prípadov máme k dispozícii výlučne zašifrovaný text, je preto potrebné určiť kategóriu šifry, ktorú nie je možné v niektorých prípadoch presne identifikovať, ale iba odhadnúť. Na základe frekvencie výskytu jednotlivých znakov v (prípade dostatočne dlhého) ZT vieme odhadnúť, že aký spôsob šifrovania bol použitý:

- Ak $|\mathcal{A}_C| > |\mathcal{A}_P|$ a frekvencia znakov je vyhladená [29], pravdepodobne sa jedná o *homofónnu substitúciu*.
- Ak $|\mathcal{A}_C| = |\mathcal{A}_P|$
  - a frekvencia znakov je vyhladená, pravdepodobne sa jedná o *polyalfabetickú substitúciu*;
  - a frekvencie znakov nie sú vyhladené (kopírujú charakteristiky jazyka, ktorú predpokladáme), pravdepodobne sa jedná o *monoalfabetickú substitúciu* alebo o *transpozičnú šifru*;

Rozdiel medzi transpozíciou a monoalfabetickou substitúciou sa dá jednoznačne určiť. V prípade transpozície by mali frekvencie jednotlivých znakov ZT priamo kopírovať charakteristiku použitého jazyka. V prípade monoalfabetickej substitúcie sa frekvencie tiež zachovávajú, len sú priradené k iným znakom (napr. znak 'd' bude mať frekvenciu znaku 'a') [29].

1.3.2 Metódy lúštenia

Využitie modernej výpočtovej sily je najvhodnejším prostriedkom pri lúštení klasických šifier, preto sa v nasledujúcej časti sústredíme výlučne na túto metodiku. Podľa stupňa počítačovej automatizácie je možné rozdeliť využitie počítačov pri kryptoanalýze na *počítačom asistovanú*, *poloautomatickú* a *automatickú* kryptoanalýzu [29].
V prvom prípade sa počítač používa ako pomôcka pri lúštení. Vykonávajú sa na ňom len pomocné výpočty. V prípade poloautomatickej kryptoanalýzy počítač prehľadáva priestor riešení, ale niektoré rozhodujúce kroky vykonáva sám kryptoanalytik. Automatická kryptoanalýza je technika, kde počítač pracuje samostatne (ako čierna skrinka - black box), kryptoanalytik obdrží až kandidátov na riešenie.
Výber vhodnej metódy na lúštenie klasických šifier je závislý od znalosti šifrovacieho algoritmu ako aj od veľkosti priestoru kľúčov (možných riešení). Metódy lúštenia klasických šifier sú založené [29] zväčša na nasledovných prístupoch:

1. *Lúštenie hrubou silou* - základom lúštenia je úplné prehľadávanie priestoru kľúčov $\mathcal{K}$. Týmto spôsobom je možné lúštiť len šifry, v prípade ktorých priestor kľúčov nie je príliš veľký. Možnosti tejto techniky značne závisia od spôsobu vyhodnotenia kľúčov (kandidátov na riešenie). Pre zefektívnenie takéhoto algoritmu je možné použiť rôzne techniky, ako napr. negatívne testy alebo rýchle vyhodnocovacie testy, vhodné na redukovanie kandidátov na riešenie [29, 32].

2. *Špecifické metódy lúštenia vychádzajúce z charakteru šifry* - pri kryptoanalýze klasických šifier je možné zostrojiť útoky, ktoré využívajú špecifické slabiny danej kategórie šifier, alebo konkrétnej šifry[3] [29]. Pre využitie týchto slabín je však nutnou podmienkou získať informácie o kategórii použitého šifrovacieho algoritmu.

3. *Lúštenie optimalizačnými algoritmami* - jedná sa o inteligentné prehľadávanie priestoru kľúčov, ktorý nie je možné prehľadať hrubou silou. Základom je ohodnotenie kvality kľúčov $k \in \mathcal{K}$ pomocou účelovej funkcie. Cieľom algoritmu je nájsť globálne optimum účelovej funkcie efektívnejšie, ako by tomu bolo v prípade prehľadávania hrubou silou [13, 29].

Pri kryptoanalýze klasických šifier je dôležitou úlohou analýza textov. Znalosť a dôkladná analýza rôznych štatistických vlastností predpokladaného[4] jazyka sa využíva pre konštrukciu funkcií na ohodnotenie kvality možných výsledkov lúštenia [32].

[3] Dobrým príkladom je útok na homofónnu Zodiacovú šifru Z408 na základe cyklického opakovania homofónov [3].

[4] Správne určenie použitého jazyka pri lúštení je vo väčšine prípadov nevyhnutné.

Kapitola 2

Analýza a ohodnotenie textu

V tejto kapitole sa zaoberáme možnosťou využitia štatistických vlastností písaného textu pri lúštení klasických šifier. Keďže pred samotným začiatkom výskumu je vždy potrebné vymedzenie skúmaných elementov, časť prvej podkapitoly bude venovaná ich charakteristike. V ďalších sekciách kapitoly podrobnejšie popíšeme vlastnosti jazyka a písaného textu (ako napr. frekvenčná charakteristika textových elementov), z ktorých sa vychádza pri vytvorení matematického popisu resp. reprezentácie jazyka, tzv. štatistického modelu jazyka. Ďalej preskúmame možnosť identifikácie zmysluplného textu v prirodzených jazykoch a uvedieme štandardné metódy na ohodnotenie textu.

2.1 Základné pojmy

Skúmaním prirodzených jazykov sa zaoberá samostatná vedná disciplína, *lingvistika*. Lingvistika sa člení na štyri základné oblasti výskumu [18] (systémová a kognitívna lingvistika, sociolingvistika a lingvistická pragmatika), z ktorých je pre ciele predkladanej práce dôležitá systémová lingvistika, zaoberajúca sa opisom a skúmaním stavby jazyka. Pre popis vlastností konkrétneho písaného textu je dôležitá tzv. *kvantitatívna charakteristika* jazyka, podstatou ktorej je frekvenčná analýza elementov textu (sekcia 2.3.1), ako aj skúmanie entropie a redundancie (sekcia 2.4.1). [18, 19, 98, 108]

Základné elementy (segmenty) jazyka tvoria presne špecifikované časti ako napr.: *foném, hláska, slovo, veta* atď., ktoré sa delia na ortografické (písané) a fonetické (hovorené) [98].
Základné ortografické elementy [98, 29, 108, 1] sú *graféma* (tvorí základnú jednotku písomného textu, napr.: písmeno), *n-gram* (n-tica susediacich grafém), *slovo* (element nesúci význam, vytvorený pomocou grafém) a *veta* (element vytvorený pomocou slov)[1].

Jednotlivé elementy jazyka vykazujú isté charakteristiky, ktoré môžu byť jednoduchšie (ako napr. percentuálna proporcia v texte), alebo komplexnejšie [108].

[1] Treba však pripomenúť, že definícia základných textových elementov sa nemusí rovnako hodiť pre všetky jazyky. Autori v [108] ako príklad uviedli európske a niektoré indiánske jazyky.

2.2 Matematický model jazyka

V prípade počítačovej reprezentácie jazyka je východiskovým bodom vytvorenie matematického popisu (tzv. modelu) jazyka, ktorý je založený na čo najpresnejšom určení charakteristík konkrétneho prirodzeného jazyka. Pomocou vytvorenia modelu jazyka je možné popísať dôležité štatistické vlastnosti, ktoré sa dajú využiť pri analýze textu. Základným spôsobom tvorby modelu jazykov pri kryptoanalýze klasických šifier sú Markovské reťazce [91, 25]. Markovský reťazec môžeme vo všeobecnosti vyjadriť ako konečný počet stavov $S_1, \ldots, S_n$, s príslušnou množinou prechodových pravdepodobností (že systém prechádza z jedného stavu do ďalšieho) [91]. *N-gramové* modely predstavujú základ Markovských štatistických modelov jazyka [100], kde pravdepodobnosti nových stavov sú podmienené pravdepodobnosťami $r = n-1$ predchádzajúcich stavov. Pravdepodobnosť postupnosti $S = S_1, \ldots, S_m$ môžeme vyjadriť ako [100]:

$$p(S) = p(S_1)p(S_2|S_1)p(S_3|S_1, S_2)\ldots p(S_m|S_1\ldots S_{m-1}) \tag{2.1}$$

Napr. pre voľbu rádu r=1 (nový stav závisí len na predošlom stave), môžeme napísať pravdepodobnosť postupnosti S ako:

$$p(S) = p(S_1)\prod_{i=2}^{m} p(S_i|S_{i-1}) \tag{2.2}$$

Vytvorenie textu v prípade prirodzeného jazyka môžeme chápať ako stochastický proces. Zmysluplný text v prípade Markovského reťazca teda predstavuje sekvenciu symbolov (alebo slov) určenú množinou pravdepodobností z diskrétneho zdroja [91]. Na základe Markovských reťazcov je teda možné aproximovať prirodzený jazyk. Príklady rôznych n-gramových aproximácií[2] uvádza C. E. Shannon v [91] :

1. Aproximácia 0. rádu - každá graféma sa vyberá rovnakou pravdepodobnosťou a nezávisle.

2. Aproximácia 1. rádu - každá graféma sa vyberá nezávisle, ale s pravdepodobnosťou, akou sa vyskytuje v prirodzenom jazyku.

3. Aproximácia 2. rádu - na základe 2-gramov. Po vybratí grafémy sa nasledujúca graféma vyberá s pravdepodobnosťou, ktorou sa vyskytuje po prvom vybratom graféme v prirodzenom jazyku.

Z pohľadu nášho výskumu predstavuje zaujímavú oblasť aproximácia 1. a 2. rádu[3] (resp. aproximácie vyšších rádov).

[2]Okrem grafém sa môže vytvoriť model na základe pravdepodobnosti slov. Konkrétne príklady jednotlivých aproximácií anglického jazyka je možné nájsť v [91] str. 7.

[3]Treba však pripomenúť, že zmenou stavov je možné previesť vyššie rády na nižšie.

2.3 Porovnanie a identifikácia textov a jazykov

Na porovnanie a identifikáciu jednotlivých jazykov a textov sa využívajú také kvantitatívne charakteristiky textových elementov, ktoré sú pre daný jazyk špecifické[4]. Pod porovnaním textov vlastne chápeme porovnanie jednej alebo niekoľkých charakteristík textových elementov [108]. Pri tom dôležitým predpokladom je tzv. ergodickosť [29]. Ergodicita vyjadruje "štatistickú homogenitu", čiže to, že získané číselné charakteristiky s rastom veľkosti zdroja sa približujú k limitným hodnotám nezávisle od konkrétneho zdroja [91]. Z toho vyplýva, že pri dostatočne dlhom texte nadobudnú číselné charakteristiky konkrétnu hodnotu a bude možné ich považovať za referenčnú hodnotu pre daný jazyk. V prípade testovania dostatočne dlhého textu by sa mali hodnoty jednotlivých špecifických charakteristík priblížiť k referenčným hodnotám [25, 50, 31, 32]. Pod výrazom dostatočne dlhý text sa chápe text takej minimálnej dĺžky, ktorý vykazuje a pri zväčšení zachováva charakteristiky jazyka. V tomto prípade je text *reprezentatívny* [108].

2.3.1 Frekvenčná charakteristika textových entít

Základnou kvantitatívnou charakteristikou jazyka je frekvenčná charakteristika elementov textu [98]. Pri lúštení klasických šifier sa najčastejšie používa frekvenčné rozdelenie rôznych n-gramov (väčšinou pre malé n, t.j. $n = 1, 2, 3$), prípadne početnosť slov. Frekvencia grafém (1-gramov) tvorí základ pre ďalšie komplexnejšie charakteristiky a indexy [29, 50, 31].

Pri stanovení hodnoty konkrétnej frekvenčnej charakteristiky sa dá využiť fakt, že "relatívna frekvencia nejakého javu pri viacnásobnom opakovaní je približne rovnaká pravdepodobnosti tohto javu" [29]. Nech náhodná premenná X_i vyjadruje výskyt znaku λ na i-tom mieste v texte ($X_i = 1$, keď i-ty znak je λ; inak je $X_i = 0$ nezávisle na i), potom platí [29]:

$$P(X_i = 1) = 1 - P(X_i = 0) = p_\lambda = E(X_i). \tag{2.3}$$

Následne pre text dĺžky n znakov platí [29]:

$$p_\lambda = E\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right). \tag{2.4}$$

Ak predpokladáme, že postupnosť náhodných premenných X_i je ergodická vzhľadom na strednú hodnotu [29] (konverguje v kvadratickom strede), tak na odhad pravdepo-

[4]Voľba konkrétnej charakteristiky (pre identifikáciu alebo porovnanie) závisí od použitého štatistického modelu jazyka.

dobnosti p_λ výskytu znaku λ môžeme použit'[5] aritmetický priemer [29]:

$$p_\lambda = \frac{1}{n}\sum_{i=1}^{n} X_i. \tag{2.5}$$

Na získanie rôznych kvantitatívnych charakteristík textových entít pre rôzne jazyky existuje množstvo zdrojov, avšak uvádzané hodnoty bývajú väčšinou buď neúplné, alebo boli vyhotovené z nie dostatočne veľkého korpusu. Pre potreby nášho výskumu sme preto experimentálne získali všetky potrebné charakteristiky (frekvencia grafém, n-gramov a slov), ktoré sme použili na všetky naše výpočty a experimenty.

Poznámka 1 *Tieto údaje boli získané z anglického OANC korpusu [2] (Open American National Corpus), ktorý obsahuje približne 79 miliónov znakov a pozostáva z textov rôznych žánrov. Zdrojový text sme previedli do telegrafnej abecedy bez medzery. Po prevedení sme dostali vzorku, ktorý obsahuje približne 56.5 miliónov znakov.*

2.3.2 Reprezentatívna dĺžka textu

Pred samotným porovnávaním frekvenčnej charakteristiky textových elementov s referenčnou hodnotou je potrebné určit' minimálnu dĺžku testovacej vzorky, ktorá bude vykazovat' charakteristiky približne zhodné s hodnotami jazyka, t.j. aby bola vzorka reprezentatívna [108]. Tejto téme sa venuje aj L. Kubáček v článku [49]. Konkrétne sa zaoberá odvodením intervalov spoľahlivosti textových entít s multinomiálnym rozdelením pomocou odhadu konfidenčného elipsoidu [49, 29].
Nech náhodná premenná X nadobúda hodnoty $x_1, \ldots, x_k$. Pravdepodobnost' nadobudnutia hodnoty $x_i = p_i > 0$ a $p_1 + \ldots + p_k = 1$, ktoré odhadujeme relatívnymi početnost'ami s normálnym rozdelením $\mathcal{N}(\mu_i, \sigma_i^2)$. Nech r je priemerná smerodajná odchýlka merania p_i (ktorú si môžeme určit' dopredu). Na stanovenie veľkosti výberu N, aby bol reprezentatívny pre náhodnú premennú X môžeme použit' vzorec [29]:

$$N = \frac{1}{r^2}\prod_{i=1}^{k} p_i^{\frac{1}{k-1}} \tag{2.6}$$

Problém však predstavuje vhodné nastavenie r, ktoré je možné určit' v niektorých prípadoch len experimentálne (napr. pre veľké k, kde jednotlivé pravdepodobnosti sú malé). V prípade malých hodnôt p_i sa preto odporúča zlúčit' málo pravdepodobné prvky ($p_i \cong 0$) do jednej skupiny. Autori v [29] uvádzajú vypočítané hodnoty N pre rôzne jazyky pre zvolené $r = 0.01, r = 0.001$ a $r = 0.005$.

[5] Iným spôsobom odhadu pravdepodobnosti textových elementov v lingvistike je napr. metóda A. Turinga a I.J. Gooda [24, 23], ktorá zahŕňa aj pravdepodobnost' textových elementov, ktoré sa v testovacej vzorke nenachádzajú.

2.3.3 Metódy porovnania frekvenčných charakteristík

Porovnanie frekvenčných charakteristík textových elementov[6] tvorí základ pre porovnanie textov. Existujú rôzne prístupy na porovnanie týchto charakteristík[7]. Ak používame pravdepodobnostné rozdelenia, potom porovnávame konečné vektory, ktorých súradnice sú z intervalu $\langle 0, 1\rangle$. Možným spôsobom je vyjadrenie podobnosti alebo vzdialenosti dvoch vektorov (frekvenčných charakteristík). V takýchto prípadoch sa väčšinou porovnáva vektor nameraných hodnôt z testovacej vzorky s vektorom očakávaných hodnôt, ktoré sa získajú z dostatočne veľkého korpusu. Na vyjadrenie podobnosti/vzdialenosti sa však dajú použiť aj funkcie, ktoré sa používajú na testovanie štatistických hypotéz. V tomto kontexte sa používa len hodnota funkcií testu bez testovania hypotéz a hľadá sa riešenie s minimálnou chybou I. druhu [51, 11, 12].
Frekvenčné charakteristiky tvoria základ aj pre špecifické metódy porovnávania textov, využívané pri kryptoanalýze klasických šifier. V tejto kapitole uvádzame niektoré základné metódy porovnania, ako aj špecifické metódy z najvýznamnejších publikácií z tejto oblasti, ako sú napr. publikácie W. F. Friedman-a [21], S. Kullback-a [50, 51] a C. E. Shannon-a [92].
Pri popise štatistických metód sa budeme držať nasledujúcej terminológie:
$\vec{X} = X_1, \ldots, X_n$ sú realizácie náhodnej premennej X, ktorá nadobúda hodnoty $1, \ldots, k$ s pravdepodobnosťami $\vec{P} = \{p_1, \ldots, p_k\}$, t.j. $P(X = i) = p_i, i = 1, \ldots, k$. Očakávanú pravdepodobnosť označujeme ako $\vec{Q} = \{q_1, \ldots, q_k\}$. Hodnoty, ktoré nadobúda náhodná premenná X niekedy nazývame aj stavy. Početnosť výskytu stavu i pri realizácii pokusu je potom np_i a jej očakávaná hodnota nq_i.

Základné štatistické vzdialenosti

Vzdialenosť vektorov predstavuje jednoduchý spôsob porovnania vzdialenosti dvoch pravdepodobnostných rozdelení [62]. Vyjadruje nakoľko sú dva objekty umiestnené v metrickom priestore vzdialené od seba [11]. Na vyjadrenie vzdialenosti je preto potrebné si zadefinovať tzv. metriku resp. metrický priestor [12]. Vo všeobecnosti na metrickom priestore nemusí byť definovaná žiadna operácia.

Definícia 3 *Nech M je neprázdna množina a nech existuje zobrazenie $d : M \times M \to \langle 0, \infty)$ spĺňajúce pre každé $x, y, z \in M$ nasledujúce vlastnosti:*

- $d(x, y) \geq 0$ *(nezápornosť)*
- $d(x, y) = 0 \iff x = y$ *(odlíšiteľnosť zhodných objektov)*
- $d(x, y) = d(y, x)$ *(symetria)*
- $d(x, z) \leq d(x, y) + d(y, z)$ *(trojuholníková nerovnosť)*

[6] Uvažujeme Markovské modely rádu n a grafémy a slová ako základné elementy.

[7] Existuje veľké množstvo funkcií na vyjadrenie vzdialenosti/podobnosti, napr. v [11] uvádzajú 45 rôznych funkcií. V tejto práci sa zameriame, len na vybranú časť najznámejších funkcií, ktoré sa používajú pri porovnávaní textov.

Potom (M, d) *nazývame metrický priestor.*

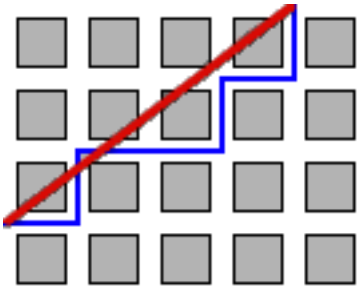
Všetky základné štatistické vzdialenosti sú odvodené od tzv. ***Minkowského*** vzdialenosti [62]. Minkowského vzdialenosť sa niekedy označuje aj ako L_c norma [11]. Je definovaná pre $c \in \langle 0, \infty)$ ako

$$MD(\vec{P}, \vec{Q}) = \left(\sum_{i=0}^{k} |p_i - q_i|^c \right)^{\frac{1}{c}}. \tag{2.7}$$

Špeciálnou voľbou parametra c dostaneme

1. pre $c = 1$ ***Manhatanskú*** vzdialenosť;
2. pre $c = 2$ ***Euklidovskú*** vzdialenosť;
3. pre $c = 1$ a malou úpravou ***štatistickú*** vzdialenosť $S(\vec{P}, \vec{Q}) = \frac{1}{2}\left(\sum_{i=0}^{k} |p_i - q_i| \right)$;
4. pre $c = \infty$ ***MAX*** vzdialenosť $MAX(\vec{P}, \vec{Q}) = \max\{|p_i - q_i|\}$;

Štatistická a MAX vzdialenosť nadobúdajú len hodnoty z intervalu $\langle 0, 1\rangle$ a sú príkladom tzv. normovanej vzdialenosti.
S rastúcim c pre MD metriku sa znižuje dopad extrémnych hodnôt (veľkých rozdielov) na celkový výsledok [62, 82] a dostaneme tzv. MAX metriku.
Vizualizácia rozdielu medzi Manhatanskou a Euklidovskou vzdialenosťou je na obrázku 2.1, kde červená čiara znázorňuje Euklidovskú vzdialenosť a modrá čiara Manhatanskú.



Obr. 2.1: Znázornenie Euklidovskej a Manhatanskej vzdialenosti
(vytvorené na základe ukážky z [94])

Manhatanská vzdialenosť je často používaná v prácach, ktoré sa zameriavajú na lúštenie klasických šifier pomocou meta-heuristík ako napr. [96, 20, 13, 16]. V prípade lúštenia klasických šifier môžeme využívať taktiež rôzne variácie štatistickej vzdialenosti, ako napr. kombinácia Manhatanskej vzdialenosti n-gramov pre rôzne n naraz, ako aj rôzne váhovanie jednotlivých kombinovaných častí (viď. [29]).

Podobnosti

Na vyjadrenie podobnosti frekvenčných charakteristík môžeme použiť aj rôzne funkcie, ktoré nespĺňajú axiómy metriky. Podobnosť vyjadruje (meria) množstvo spoločných informácií, ktoré zdieľajú dva objekty [12]. Na rozdiel od metrických vzdialeností je väčšina podobnostných funkcií normalizovaná na interval $\langle 0, 1\rangle$, kde 1 vyjadruje 100% podobnosť[8] [11].

Definícia 4 *Nech M je neprázdna množina a nech existuje zobrazenie $s : M \times M \to \langle 0, \infty)$ spĺňajúca pre každé $x, y, z \in M$ nasledujúce vlastnosti:*

- $s(x, x) \geq 0$
- $s(x, x) \geq s(x, y)$
- $s(x, y) = s(y, x)$
- $s(x, y) + s(y, z) \leq s(x, z) + s(y, y)$
- $s(x, x) = s(y, y) = s(x, y) \iff x = y$

Potom (M, s) nazývame symetrický priestor.

V prípade normalizácie[9] metriky je možná transformácia danej vzdialenosti $d(x, y)$ na podobnosť $s(x, y)$ [12] nasledovne:

$$s(x, y) = 1 - d(x, y). \tag{2.8}$$

Jednou z mier podobnosti dvoch textov je využitie ***kosínusovej podobnosti*** [42, 84, 73]. Porovnáva sa uhol medzi dvoma vektormi namiesto ich veľkosti [73]. Výsledná hodnota vyjadruje koreláciu medzi dvomi vektormi [39]. V prípade, že porovnávame relatívne frekvencie textových elementov a výsledok interpretujeme priamo uhlom medzi vektormi, kosínusová podobnosť spĺňa axiómy metriky a môžeme ju zaradiť medzi metódy vyjadrenia vzdialenosti [77]. Vo všeobecnosti sa však zaraďuje do skupiny podobností.

Kosínusovú podobnosť dvoch vektorov $\vec{P}$ a $\vec{Q}$ môžeme vyjadriť ako kosínus uhla medzi danými vektormi [10, 73, 42]:

$$CS(\vec{P}, \vec{Q}) = \frac{\sum_{i=0}^{k} p_i q_i}{\sqrt{\sum_{i=0}^{k} p_i^2} \sqrt{\sum_{i=0}^{k} q_i^2}}. \tag{2.9}$$

[8] Táto vlastnosť je výhodnejšia ako vzdialenosť vo viacerých oblastiach.
[9] Pod normalizáciou chápeme prevedenie na interval $\langle 0, 1\rangle$.

Vzdialenosť od nezávislosti

Často používanou metódou porovnania podobnosti dvoch frekvenčných charakteristík je ***Pearsonova*** χ^2 ***divergencia*** (ďalej len χ^2 divergencia) [60, 25, 56, 81]. Ako je známe, táto testovacia funkcia vyjadruje do akej miery sa výsledky nezávislých pokusov (početnosti v jednotlivých kategóriách) podobajú očakávaným početnostiam [60]. χ^2 divergencia sa ráta podľa vzťahu (2.10). Keďže nespĺňa požadované axiómy vzdialenosti ani podobnosti, zaraďuje sa podľa ich významu medzi štatistické testy nezávislosti.

$$\chi^2(\vec{P}, \vec{Q}) = \sum_{i=1}^{k} \frac{(np_i - nq_i)^2}{nq_i} \tag{2.10}$$

Medzi komplexnejšie metódy vyjadrenia podobnosti pravdepodobnostných rozdelení patrí napr. ***Jensen-Shannonova divergencia*** [101, 54] (ďalej len JSD), ktorá je založená na Jensenovej nerovnosti a Shannonovej entropii [54]. V niektorých publikáciách ako [101] nazývajú túto metódu ako symetrizovanú verziu Kullback-Leibler divergencie.

JSD (divergenciu $\vec{P}$ od $\vec{Q}$) môžeme vypočítať pomocou vzorca (2.11) [101] ako:

$$JSD(\vec{P}, \vec{Q}) = \frac{1}{2}KL(\vec{P}, \vec{M}) + \frac{1}{2}KL(\vec{Q}, \vec{M}), \tag{2.11}$$

kde:

$$\vec{M} = \frac{1}{2}(\vec{P} + \vec{Q}), \tag{2.12}$$

a $KL(\vec{P}, \vec{Q})$ je Kullback-Leibler divergencia [54, 34]:

$$KL(\vec{P}, \vec{Q}) = \sum_{i=1}^{k} p_i \log_2 \frac{p_i}{q_i}. \tag{2.13}$$

Alternatívny spôsob vypočítania JSD je pomocu Shannonovej entropie [54] (sekcia 2.4.1).
Nech $\vec{P}, \vec{Q}$ sú pravdepodobnostné rozdelenia s váhami $w_1, w_2 \geq 0, w_1 + w_2 = 1$:

$$JSD_w(\vec{P}, \vec{Q}) = H(w_1\vec{P} + w_2\vec{Q}) - w_1H(\vec{P}) - w_2H(\vec{Q}). \tag{2.14}$$

Iné spôsoby porovnania

Index koincidencie [22, 50] (ďalej len IC) je založený na relatívnych frekvenciách textových elementov a vyjadruje pravdepodobnosť javu, že 2 náhodne vybraté textové elementy budú totožné [25, 29]. Vychádza z Φ-testu od W. Friedmana z publikácie [21]. Φ-test sa využíva na identifikáciu jazyka v prípade monoalfabetickej substitúcie (kde hodnota Φ-testu je odlišná pre rôzne jazyky)[29].

Hodnota Φ-testu sa vypočíta ako:

$$\Phi = \sum_i np_i(np_i - 1). \tag{2.15}$$

IC predstavuje len normalizáciu Φ-testu [29, 34], a vypočíta sa ako:

$$IC = \frac{1}{n(n-1)} \sum_i np_i(np_i - 1) \approx \sum_i p_i^2. \tag{2.16}$$

Výsledná hodnota IC sa môže pohybovať medzi $\frac{1}{n}$ a 1, kde minimálnu hodnotu dosiahneme v prípade rovnomerného rozdelenia [34].

Alternatívny spôsob využitia IC je meranie tzv. "vyhladenosti" [34] (measure of roughness - MR). Jedná sa o vypočítanie Euklidovskej vzdialenosti frekvencií písmen od rovnomerného rozdelenia.

$$MR_n = IC - \frac{1}{n} = \sum_{i=1}^{n} (p_i - \frac{1}{n})^2 \tag{2.17}$$

Autori v [34] tiež uvádzajú možnosť rozšírenia IC na vyššie rády. IC_α rádu α (pravdepodobnosť, že α náhodne vybratých znakov bude rovnakých) sa vypočíta ako:

$$IC_\alpha = \sum_i p_i^\alpha. \tag{2.18}$$

Ďalším spôsobom využitia IC je meranie pravdepodobnosti výberu α rovnakých textových entít, iných ako grafémy, napr: n-gramy. Čiže IC môžeme zovšeobecniť na vyrátanie pravdepodobnosti, že α náhodne vybratých n-gramov bude rovnakých (p_i je teraz pravdepodobnosť $n-$gramu):

$$IC_\alpha^n = \sum_i p_i^\alpha. \tag{2.19}$$

Index koincidencie je možné využiť aj na ďalšie účely [25, 29], ako napr. identifikáciu a rozoznanie jazykov; zistenie periódy polyalfabetickej substitúcie (detailnejšie popísané napr. v [29]); identifikáciu typu šifrovacieho algoritmu.

2.4 Špecifické vlastnosti jazyka

Z hľadiska kryptoanalýzy, jazyky disponujú istými významnými charakteristikami, ako *entropia*, *redundancia* a *unicity distance*, na základe ktorých je možné stanoviť zmysluplnosť lúštenia konkrétnej šifry. Konkrétne, či výsledok lúštenia môžeme s istotou považovať za jediné správne riešenie. Pri lúštení klasických šifier je potrebné mať na zreteli vyrátanú hodnotu unicity distance pre danú šifru, z ktorej vyplýva, či je dĺžka sledovaného ZT dostatočne dlhá na to, aby sme sa dopracovali k správnemu výsledku.

2.4.1 Entropia a redundancia jazyka

Entropiu v jednoduchšom ponímaní môžeme chápať ako množstvo informácií, ktoré získame pri prijatí správy. V určitom zmysle entropia vyjadruje neurčitosť zdroja [91]. Ako prvý sa analýzou entropie jazykov zaoberal C. E. Shannon [93, 91, 92]. V neskorších publikáciách z tejto oblasti preto entropiu označujú aj ako Shannonova entropia. V tejto práci sa zameriavame na entropiu jazykov v kontexte písaného textu.
Nech X je náhodná premenná definovaná na $\mathcal{P}$ a pravdepodobnosť výskytu grafémy λ v texte je $P(X = \lambda) = p_\lambda$, potom množstvo informácií, ktorú získavame pri konkrétnej realizácií náhodnej premennej X, je $I(X_\lambda) = -\log_2 P(X = \lambda)$. Pri viacnásobnej realizácií náhodnej premennej získavame priemerný počet informácie $H(X)$ (strednú hodnotu náhodnej premennej), ktorú nazývame entropia náhodnej veličiny [91, 29]:

$$H(X) = E(I) = -\sum_{i=1}^{k} p_i \log_2 p_i. \tag{2.20}$$

Platí pri tom ohraničenie entropie pre k možných výsledkov náhodnej premennej X [29]:

$$0 \leq H(X) \leq \log_2 k. \tag{2.21}$$

Entropiu jazyka $\mathcal{H}$ nie je možné priamo vyrátať, ale je možné aproximovať napr. pomocou štatistiky n-gramov (sekvencií dĺžky n) [91]. Ako zdroj správy môžeme uvažovať napr. Markovské zdroje r-tého rádu, t.j. písmeno závisí od r predchádzajúcich písmen [30]. V takomto prípade sa predpokladá stacionarita zdroja (viď. [30], str. 48-50) ako aj homogenita náhodnej postupnosti [30]. Entropia pre stacionárny Markovský zdroj r-tého rádu je daná vzťahom (2.22) a (2.23) [30].

$$\mathcal{H} = H(X_{r+1}/X_1, X_2, \ldots, X_r) \tag{2.22}$$

$$\frac{1}{n} H(X_1, X_2, \ldots, X_n) \geq \frac{n-r}{n} \mathcal{H} \tag{2.23}$$

Shannon vo svojej práci [91] uvádza spôsob aproximácie entropie zdroja pomocou podmienenej entropie F_n (vzťah (2.24)) (entropia grafémy po znalosti n-1 predchádzajúcich).

$$F_n = -\sum_{i,j} p_{i,j} \log_2 p_{i,j} + \sum_i p_i \log_2 p_i. \tag{2.24}$$

Vyrátanie F_n pre malé hodnoty n je pomerne jednoduché, pomocou frekvencii grafém, 2-gramov a 3-gramov. Shannon vo svojej práci uvádza nasledovné vyrátané hodnoty pre anglický jazyk [93]:

- $F_1 = -\sum_{i=1}^{26} p_i \log_2 p_i = 4.14$ bitov;
- $F_2 = -\sum_{i,j} p_{i,j} \log_2 p_{i,j} + \sum_i p_i \log_2 p_i = 7.70 - 4.14 = 3.56$ bitov;

- $F_3 = -\sum_{i,j,k} p_{i,j,k} \log_2 p_{i,j,k} + \sum_{i,j} p_{i,j} \log_2 p_{i,j} = 11.0 - 7.70 = 3.3$ bitov.

Shannon-ovú entropiu pre anglický jazyk odhadujú na $\mathcal{H}_{AJ} = 1.5$ bitov na znak [63].

Experimentálne sme vyrátali entropiu zdroja pre anglický jazyk, pre Markovský model do 8-ho rádu podľa pôvodného Shannonovho vzťahu (2.24). Výsledné hodnoty entropie zdroja sú uvedené tabuľke 2.1. Na testy sme použili korpus bez medzery. Získané aproximácie Shannon-ovej entropie zhruba zodpovedajú údajom z [74], kde testy boli vykonané na rovnakom OANC korpuse.

n	0	1	2	3	4	5	6	7	8
H (2.20)	4.7	4.2	7.8	11.1	13.9	16.3	18.3	19.9	21.3
F_n (2.24)	4.7	4.2	3.7	3.3	2.8	2.4	2.0	1.7	1.4

Tabuľka 2.1: Aproximácie rôznych rádov (pre $n = 1, \ldots, 8$) Shannonovej entropie a relatívnej entropie (AJ bez ⊔)

Entropia úzko súvisí s pojmom ***redundancia*** [91]. Štatistická štruktúra jazyka spôsobuje istú nadbytočnosť niektorých elementov písaného textu, čo sa nazýva redundancia [93]. Redundanciu spôsobujú napríklad najfrekventovanejšie grafémy, alebo grafémy, ktoré sa vyskytujú vo dvojici s inou konkrétnou grafémou s veľmi vysokou pravdepodobnosťou (napr. graféma *e* alebo dvojica *qu* v anglickom jazyku).

Redundanciu textu[10] (R) môžeme vypočítať pomocou relatívnej entropie nasledovne [91]:

$$R = 1 - H_r = 1 - \frac{H}{H_{max}}, \tag{2.25}$$

kde relatívna entropia (H_r) vyjadruje pomer entropie zdroja H a maximálnej entropie H_{max}, kde maximálna entropia nastane v prípade rovnakých pravdepodobností výskytu grafém [91] (viď. vzťah (2.21)), čiže $p_i = \frac{1}{k}$ pre $\forall i \in \{1, 2, \ldots, k\}$. Výsledná hodnota vzťahu (2.25) je vyjadrená v % (z intervalu $\langle 0, 1 \rangle$).
Redundanciu môžeme vyjadriť aj v bitoch na znak ($\mathcal{R}$) ako rozdiel maximálnej entropie a entropie jazyka. Čiže v tomto prípade dostaneme, že o koľko bitov sa použije viac, ako je potrebné na zápis znaku:

$$\mathcal{R} = H_{max} - H. \tag{2.26}$$

Vyrátané hodnoty entropie a redundancie z rôznych slovenských literárnych diel je možné nájsť v publikáciách [47, 48]. Pre anglický jazyk sú uvedené aproximácie entropie pre vyššie rády v [33] a na internetovej stránke [74].

[10] Shannon v [91] uvádza, že redundancia anglického jazyka je zhruba 50%, čiže polovica písmen v texte je definovaná štruktúrou jazyka a polovica je vybratá "voľne".

2.4.2 Unicity distance

Shannon v [92] uviedol zaujímavú otázku pri skúmaní entropie a redundancie jazyka v oblasti kryptoanalýzy - a to tú, že aká je minimálna dĺžka zašifrovanej správy, aby počet tzv. falošných kľúčov bol nulový. Inak povedané, či po dešifrovaní správy (napr. pomocou hrubej sily) dostaneme jediné zmysluplné riešenie.
Zoberme si prípad kryptosystému (Definícia 1), kde $|\mathcal{P}| = |\mathcal{C}|$ a každý kľúč sa použije s rovnakou pravdepodobnosťou. Nech $X \in \mathcal{P}$, $Y \in \mathcal{C}$ a $K \in \mathcal{K}$, množstvo informácie ohľadom kľúča, ktorú vieme získať na základe zašifrovanej správy je $H(K|Y)$, pri čom platí [92, 28]:

$$H(K|Y) = H(K) + H(X) - H(Y) \geq H(K) - \mathcal{R} \tag{2.27}$$

Vzťah (2.27) teda vyjadruje množstvo informácie, ktorú získavame navyše zo zašifrovaného textu. Je zrejmé, že v prípade ideálnych podmienok ($H(X) = H(Y)$) by sme nemali byť schopný získať žiadne dodatočné informácie, t.j. každý nový zachytený znak zo ZT, by mal uchovať maximálnu neurčitosť. Problém však predstavuje samotná štruktúra - nadbytočnosť jazyka. To znamená, že napriek tomu, že nepoznáme OT, na základe štruktúry jazyka máme dodatočné informácie ($H(X) - \mathcal{R}$) . Čiže redundancia jazyka $\mathcal{R}$ nám znižuje neurčitosť jazyka. Aby sme odstránili dôsledky redundancie (aby sme dosiahli $H(K|Y) = 0$), potrebujeme mať správu minimálne dĺžky U (podľa vzťahu (2.28)).

$$U\mathcal{R} \geq H(K) \tag{2.28}$$

Táto hodnota U sa nazýva **unicity distance** [92] a vyjadruje hodnotu, pri ktorej počet falošných kľúčov je 0 [28]. Po úprave (2.28) ho vypočítame ako:

$$U \approx \frac{H(K)}{\mathcal{R}} \tag{2.29}$$

Autori v [28] uvádzajú presnejšie vyjadrenie hodnoty *unicity distance* na základe rádu aproximácie entropie zdroja $\mathcal{H}$. V [78] uvádzajú *unicity distance* pre monoalfabetickú substitúciu.

2.5 Ďalšie spôsoby ohodnotenia textov

Na ohodnotenie textu môžeme použiť aj metódy, ktoré nie sú priamo založené na porovnávaní kvantitatívnych charakteristík jazyka. Jedná sa o alternatívne spôsoby ohodnotenia zmysluplnosti textu, alebo o spôsoby vyradenia nie zmysluplných textov. Jedným spôsobom vyjadrenia zmysluplnosti je využitie ohodnocovacích metód pomocou **slovníkov**. Táto metóda je založená na predpoklade, že v prípade zmysluplného textu sme schopní pokryť text slovami z referenčného slovníka. Úspešnosť tejto metódy však značne závisí na kvalite stanoveného referenčného slovníka [31]. Metódy založené na slovníkovom ohodnotení sa využívajú hlavne na ohodnotenie krátkych textov, kde štatistické vlastnosti jazyka nie sú reprezentatívne pre zvolený model [69, 35].
Inou alternatívou sú metódy, ktoré slúžia na odfiltrovanie textov, ktoré obsahujú málo pravdepodobné alebo nie zmysluplné úseky [32, 31].

Uvedené spôsoby ohodnotenia často slúžia ako dodatok pre štandardné metódy založené na porovnávaní kvantitatívnych charakteristík [31] uvedené v sekcii 2.3.3.

2.5.1 Slovníkové ohodnotenie

Slovo je najmenším elementom jazykovej komunikácie, ktoré je nositeľom významu [1]. Zmysluplnosť textu sa teda dá vyjadriť prostredníctvom slov, nosiacich význam, čiže analýzou slov v texte. Analýza prebieha pomocou vybratej množiny slov z referenčného jazyka, tzv. slovníka. Tieto metódy sa preto nazývajú **slovníkové ohodnotenie** a predstavujú pomerne presný spôsob ohodnotenia textu pri dostatočne veľkom slovníku [29].
Vo väčšine prípadov slovníkového ohodnotenia sa nejedná o priame porovnávanie kvantitatívnych charakteristík slov, ale o analýzu výskytu slov v testovacej vzorke [29, 69]. Ohodnotenie textu môže byť založené na počte nájdených slov alebo častí slov zo slovníka v testovacej vzorke [29]. Vychádza sa z predpokladu, že vysoký počet nájdených slov naznačuje zmysluplnosť textu. Treba však pripomenúť, že okrem počtu je potrebné brať do úvahy aj dĺžku jednotlivých slov [108]. Predpokladá sa, že čím dlhšie slová sa dajú nájsť v testovacej vzorke, tým väčšia časť je zmysluplná. Zmysluplnosť textu na základe slovníkového ohodnotenia teda môžeme presnejšie zadefinovať ako *nájdenie čo najväčšieho počtu čo najdlhších slov v texte*. Preto, v prípade slovníkových ohodnotení je výhodnejšie redukovať slovník na dlhšie slová, alebo hodnotiť dlhšie slová väčším skóre, napr. pomocou vzorca pre text X dĺžky L, kde N_d je počet d-písmenových slov [29, 31]:

$$F(X) = \frac{1}{L} \sum_{d=3}^{10} d^2 N_d. \tag{2.30}$$

Slovníkové ohodnotenia môžeme použiť aj ako vyjadrenie percentuálneho pokrytia textu platnými slovami z jazyka. Čím väčšiu časť textu vieme popísať týmto spôsobom, tým viac významu môžeme priradiť k testovacej vzorke.

Použitie slovníkov však má svoje obmedzenia. Vykonanie analýzy na základe veľkého slovníka je z časovej zložitosti nevýhodné [31]. Preto sa vo väčšine prípadov používa redukovaný slovník, ktorý obsahuje zvyčajne najfrekventovanejšie slová jazyka [35]. V prípade použitia menších slovníkov však väčšinou nie je možné pokryť celý text slovami.

Ďalší problém predstavuje výskyt možných gramatických chýb v testovacej vzorke. S týmto problémom súvisí aj iný pohľad na využitie slovníkov pri ohodnotení textov. V prípade kryptoanalýzy klasických šifier môžeme riešiť problém ohodnotenia čiastočné správnych textov (ktoré napríklad dostaneme po automatickom počítačovom lúštení), alebo texty ktoré obsahujú už spomenuté gramatické chyby alebo neúplné slová. V takýchto prípadoch je možné použiť rôzne techniky na kontrolu pravopisu (tzv. spell checking) alebo na priamu opravu týchto chýb. Často sú tieto metódy založené na meraní editačnej vzdialenosti reťazcov (string edit distance).
Najčastejšie použitá editačná vzdialenosť reťazcov je Levensstheinova vzdialenosť,

ktorá vyjadruje minimálny počet základných editačných operácií, potrebných na transformáciu jedného reťazca na druhý reťazec. Základné editačné operácie sú: vkladanie nového písmena, vymazanie písmena, substitúcia písmena. Táto vzdialenosť spĺňa axiómy metriky [67].
V niektorých prípadoch sa využívajú len vybraté editačné operácie. Napr. keď sa uvažuje len o substitúcií znakov, hovoríme o Hammingovej vzdialenosti reťazcov [67].

2.5.2 Negatívne filtre

Existujú prípady, keď pomocou zložitejších testov hodnotíme zmysluplnosť textov z veľkej testovacej množiny (ako napr. v prípade útokov pomocou hrubej sily). V týchto prípadoch je výhodné namiesto ohodnotenia výpočtovo zložitými metódami odfiltrovať nie zmysluplné texty, a to rýchlym a spoľahlivým spôsobom [31]. Na to slúžia metódy nazývané ako negatívne filtre [32, 31].
Negatívne filtre môžeme skonštruovať napr. na hľadanie n-gramov, ktoré sa v jazyku nevyskytujú. V tomto prípade sa z použitých n-gramov zostrojí tabuľka a podľa pravdepodobnosti výskytu daného n-gramu sa priradí buď pozitívne, alebo negatívne hodnotenie [29, 31]. Ďalším spôsobom je stanovenie maximálnej dĺžky úseku za sebou idúcich grafém v texte, ktoré sú samo- alebo spoluhlásky.

If you do not know how, observe the natural phenomena, they will give you clear answer and inspiration.

N. Tesla

Kapitola 3

Meta-heuristiky

Táto kapitola sa zoberá optimalizačnými algoritmami, nazývané aj meta-heuristiky. Pod optimalizáciou budeme chápať hľadanie lokálnych (alebo) globálnych extrémov účelovej funkcie. Na začiatku kapitoly si zadefinujeme prvky optimalizačného problému a uvedieme klasifikáciu meta-heuristík na základe rôznych charakteristík. Ukážeme princíp ich fungovania a potenciál využitia pri lúštení ťažkých optimalizačných problémov.

3.1 Základná charakteristika

Slovo ***heuristika*** je odvodené z gréckeho slova heuriskein ($\epsilon\upsilon\rho\iota\sigma\kappa\epsilon\iota\nu$), ktorého význam je vyhľadať [6]. Predpona "meta"[1] znamená "na" , "na vyššej úrovni" [27].
Vo všeobecnosti sa heuristika chápe ako efektívna a rýchla metóda pre riešenie ťažkých problémov, ktorých riešenie je časovo a výpočtovo náročné. Cieľom heuristických techník je často len aproximácia riešenia, čiže sa nejedná o nájdenie konkrétneho riešenia daného problému, ale o nájdenie takého riešenia, ktoré dostatočne vyhovuje daným požiadavkám. Ide pri tom o získanie vhodného riešenia efektívnejšie ako štandardnými metódami ako sú napr. priame metódy prehľadávania celého priestoru riešenia (tzv. brute-force) [90, 68, 106, 58].

Rozdiel medzi heuristikou a meta-heuristikou sa dá ozrejmiť nasledovne:
Meta-heuristiky nie sú problémovo závislé, čiže nie je známy konkrétny problém a ani detaily prehľadávania. Je to všeobecná stratégia riešenia optimalizačných problémov. Príkladom meta-heuristiky je procedúra (metóda) schopná riešiť rôzne triedy optimalizačných úloh. V prípade, že pomocou takejto metódy riešime konkrétny problém, čiže poznáme parametre a detaily prehľadávania, jedná sa o heuristiku (algoritmus). V tomto prípade teda samotná idea predstavuje meta-heuristiku a jeho použitie na konkrétny problém heuristiku.

[1] Výraz meta-heuristika bol prvý krát použitý v roku 1986 [26].

Pojem ***meta-heuristika*** sa dá chápať aj ako všeobecný algoritmus, ktorý je možné aplikovať s minimálnymi úpravami na rôzne triedy optimalizačných problémov. Výhodou týchto metód je teda ich efektivita a všeobecnosť. Blum [6] charakterizuje vlastnosti meta-heuristík nasledovne:

- Meta-heuristiky sú stratégie na nasmerovanie prehľadávacích procesov.
- Ich cieľom je efektívne prehľadávanie a aspoň priblíženie sa k optimálnemu riešeniu.
- Tieto algoritmy sú aproximačné a väčšinou nedeterministické.
- Väčšinou sa využívajú techniky proti uviaznutiu v lokálnych optimách.

Keďže meta-heuristiky sa využívajú na hľadanie riešenia optimalizačných problémov, je potrebné aby sme vymedzili a zovšeobecnili pojem optimalizačného problému. Základnými prvkami optimalizačného problému [6, 79] $P = (\Theta, \mathcal{F})$ sú:

- Nech D_i sú prípustné hodnoty pre súradnice vektora $(x_1, \dots x_n), x_i \in D_i$. Nech $\Theta = \{(x_1, \dots x_n), x_i \in D_i\}$
- Nech kandidát na riešenie je $\theta \in \Theta$.
- Pod susednými riešeniami pre kandidáta θ rozumieme akúkoľvek množinu $\mathcal{N}(\theta) \subseteq \Theta$ s podmienkou $\theta \in \mathcal{N}(\theta)$.
- Účelová funkcia $\mathcal{F} : \Theta \to \mathbb{R}$, kde $\mathcal{F}$ slúži na vyhodnotenie kvality $\theta \in \Theta$.

Úlohu optimalizácie môžeme vyjadriť ako nájdenie globálneho maxima funkcie $\mathcal{F}$ (t.j maximalizačný problém) [52, 79]:

$$\theta_{opt} = arg \max_{\theta \in \Theta} \mathcal{F}(\theta) \tag{3.1}$$

Poznámka 2 *Maximalizačný problém môžeme pretransformovať [90] na minimalizačný ako:* $\max(\mathcal{F}(\theta)) = \min(-\mathcal{F}(\theta))$.

Vo všeobecnosti je možné definovať optimalizáciu ako hľadanie najlepšieho riešenia pre problémy, kde kvalitu jednotlivých riešení môžeme vyjadriť číslom - hodnotou účelovej funkcie [52]. Serafino [90] popisuje pojem optimalizácie ako matematické techniky a algoritmy používané na identifikáciu a nájdenie extrémov príslušných účelových funkcii (nazývané aj ako fitnes funkcia, alebo skóre funkcia).

Pre úplné zadefinovanie štruktúry meta-heuristických metód teda je potrebné vymedziť pojmy ako lokálne maximum a globálne maximum funkcie [79, 40].

Definícia 5 *Funkcia $\mathcal{F}$ má lokálne maximum v $\theta' \in \Theta$, ak platí, že $\mathcal{F}(\theta_k) \leq \mathcal{F}(\theta')$ pre $\forall \theta_k \in \mathcal{N}(\theta')$. Čiže všetky susedné riešenia k θ' majú horšie ohodnotenie.*

Definícia 6 *Funkcia $\mathcal{F}$ má globálne maximum v $\theta' \in \Theta$, ak platí, že $\mathcal{F}(\theta_k) \leq \mathcal{F}(\theta')$ pre $\forall \theta_k \in \Theta$.*

Cieľom meta-heuristík je vo všeobecnosti dosiahnutie globálneho maxima účelovej funkcie. Počas prehľadávania priestoru riešenia však tieto metódy môžu naraziť na lokálne maximá účelovej funkcie, kde hrozí uviaznutie. Počet lokálnych maxím a náchylnosť uviaznutia meta-heuristík v lokálnych maximách závisí od účelovej funkcie $\mathcal{F}$ a od susedných riešení. Pomocou týchto prvkov môžeme zadefinovať tzv. **fitness landscape** [79], pomocou ktorej môžeme presnejšie charakterizovať (popísať vlastnosti) prehľadávací priestor.
Fitness landscape $\mathcal{L}$ môžeme zadefinovať ako trojicu $\mathcal{L} = (\Theta, \mathcal{F}, m)$, kde funkcia m je nejaká zvolená metrika (vzdialenosť dvoch riešení). V niektorých prípadoch môžeme priamo merať vzdialenosť na základe susedných riešení. V takomto prípade môžeme upresniť definíciu susedných riešení: $x \in \mathcal{N}(y) \Leftrightarrow m(x, y) \leq h$ pre $x, y \in \Theta$. Častá je voľba $h = 1$. Podľa potreby sa však môže použiť aj iná metrika, ktorá priamo nesúvisí so susednými riešeniami. [79]

Procedúra 1 popisuje základné lokálne prehľadávanie (všeobecnú štruktúru meta - heuristík) [68, 40], ktoré predstavuje iteratívny posun medzi možnými riešeniami prehľadávacieho priestoru. Tento posun (prehľadávacia trajektória) môže byť popísaný ako prechádzanie medzi možnými susednými riešeniami [40]. Treba však poznamenať, že časť meta-heuristík pracuje s jedným kandidátom $\theta^c \in \Theta$ na riešenie. Iné typy meta-heuristík pracujú s množinou riešení naraz, $\mathtt{P}_k = \{\theta^c_1, \ldots, \theta^c_j\}$, $\mathtt{P}_k \subset \Theta$. [68]

Procedúra 1: Všeobecná štruktúra meta-heuristík

1: Inicializácia počiatočného kandidáta na riešenie θ_k a inicializácia kroku $k \leftarrow 0$.
2: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia(í).
3: Výber náhodného prvku $\theta^c \in \mathcal{N}(\theta_k)$ zo susedov.
4: Akceptovanie nového riešenia $\theta_{k+1} \leftarrow \theta^c$, alebo jeho odmietnutie $\theta_{k+1} \leftarrow \theta_k$ podľa preddefinovaných pravidiel.
5: Ďalší krok $k \leftarrow k + 1$.
6: Kým nie je splnená ukončovacia podmienka, skočiť na bod 2.

Keďže meta-heuristické metódy sa používajú na prehľadávanie veľkého priestora možných riešení, je potrebné si zadefinovať potrebný čas na **ukončenie behu programu**, alebo minimálnu kvalitu riešenia pri ktorom sa výpočet zastaví. V prípade absencie ukončovacej podmienky by boli tieto metódy len variantou prehľadávania hrubou silou a stratili by svoj význam.
Ukončovaciu podmienku je možné definovať rôznymi spôsobmi [88]:

- Konštantný počet krokov, ktoré je potrebné vykonať.
- Testovanie vopred zadefinovaných podmienok, pri splnení ktorých sa výpočet ukončí.

- Nájdenie globálneho (alebo lokálneho) extrému účelovej funkcie.

Tieto podmienky je však nutné vo väčšine prípadov rozšíriť. Prípadné zaseknutie metódy v lokálnych extrémoch môže spôsobiť, že algoritmus nikdy nesplní vopred zadefinované podmienky, alebo nedosiahne určenú hodnotu globálneho extrému. V tom prípade je potrebné sledovať zmenu hodnoty účelovej funkcie. Keď sa táto hodnota nezmení (alebo je len minimálna zmena) počas vopred zadefinovaného počtu krokov, beh programu sa zastaví, prípadne sa reinicializuje a opätovne spustí.
V situáciach, keď presná očakávaná hodnota účelovej funkcie nie je vopred známa (iba sa odhaduje), zadefinuje sa minimálna požiadavka na hodnotu účelovej funkcie, ktorú chceme dosiahnuť [88, 75].

Úspešnosť meta-heuristických metód závisí od nastavenia parametrov danej metódy. Priestor možných nastavení parametrov danej metódy je vo veľkej väčšine prípadov príliš veľký na priame odhadnutie najvhodnejšieho nastavenia. Všeobecne môžeme považovať tento problém sám o sebe inštanciou optimalizačného problému [112], rovnako ako aj voľbu samotnej meta-heuristiky na riešenie daného problému. Myšlienkou vhodného nastavenia parametrov meta-heuristiky (napr. v prípade genetických algoritmov) pomocou inej meta-heuristiky sa zaoberali napr. aj Eiben, Davis a Rachenberg [8]. Použitie meta-heuristiky na nastavenie, riadenie alebo prehľadávanie priestoru meta-heuristík sa nazýva ***hyper-heuristika*** [9, 71, 8, 59].

3.2 Klasifikácia

Meta-heuristické metódy sa klasifikujú na základe rôznych špecifických charakteristík. Niekoľko možných uhlov pohľadu na spôsob klasifikácie uvádza Ch. Blum a A. Roli [6]:

- Na základe pôvodu metód
  - *prírodou neinšpirované*
  - *prírodou inšpirované*
- Na základe mohutnosti množiny kandidátov na riešenie
  - *jednoprvkové* - pracujúce s jedným kandidátom riešenia $\{\theta^c\}$
  - *populačné* - pracujúce s množinou kandidátov riešenia $\{\theta^c_1, \ldots, \theta^c_n\}$
- Podľa spôsobu použitia účelovej funkcie $\mathcal{F}$
  - *statické* $\mathcal{F}$ - účelová funkcia je nemenná počas behu programu
  - *dynamické* $\mathcal{F}$ - meniaca sa účelová funkcia počas prehľadávania
- Podľa štruktúry susedov $\mathcal{N}$

  - *jediná štruktúra susedov* - nemení sa štruktúra $\mathcal{N}$
  - *rôzne štruktúry susedov* - je možnosť vymenenia tzv. fitness landscape

- podľa použitia pamäte
  - *s pamät'ou* - použitie krátkodobej a dlhodobej pamäte
  - *bez pamäte* - na základe Markovovského procesu, používajú informácie len z aktuálneho kandidáta na riešenie

V literatúre sa uvádza aj iná zjednodušená kategorizácia týchto metód (napr. T. Weise v [106]), kde sa tieto metódy rozdeľujú na dve hlavné skupiny: *deterministické* a *pravdepodobnostné*. V d'alších častiach tejto práce však ostávame pri kategorizácii Ch. Bluma a A. Roli, o deterministických metódach nebudeme uvažovať v rámci meta-heuristík.

V nasledujúcich častiach popíšeme niekoľko základných meta-heuristických algoritmov, ako aj niekoľko vybratých z kategórie prírodou inšpirovaných. Popíšeme aj princíp ich fungovania a zaradíme ich do príslušných kategórií. Keďže niektoré výpočtové časti meta-heuristík (napr. ukončovacia podmienka[2], účelová funkcia a spôsob akceptovania nových kandidátov) je možné popísať až po špecifikácii optimalizačného problému, uvádzame len všeobecné postupy popísané pomocou procedúr.

3.3 Základné meta-heuristiky

Základnú skupinu prehľadávacích meta-heuristík tvoria horolezecký algoritmus (HC), simulované žíhanie (SA) a tabu search (TS). V prípade SA a TS sa jedná o miernu modifikáciu horolezeckého algoritmu, preto ich môžeme zovšeobecniť pomocou HC.

3.3.1 Horolezecký algoritmus

Horolezecký algoritmus je jeden z najstarších prehľadávacích meta-heuristík, ktorý pracuje s jedným kandidátom riešenia. V každej iterácií prijíma len nové riešenia s lepším hodnotením ako aktuálne riešenie [55, 52].
Zásadným nedostatkom tejto metódy je jej náchylnosť na uviaznutie v lokálnych extrémoch. Táto metóda je vhodná v prípade riešenia problému s malým počtom lokálnych extrémov. [106]
Jednoduchšou variantou horolezeckého algoritmu je *náhodné prehľadávanie* (random search - RS) [55, 52]. Rozdiel oproti HC je len v 3. bode procedúry, kde sa namiesto $\mathcal{N}(\theta_k)$ vyberá náhodne z Θ. Principiálne je táto metóda odolná voči uviaznutiu v lokálnych extrémoch, ale stratí sa pri tom riadenie prehľadávania a tým aj konvergencia ku globálnemu optimu.

[2]Vo väčšine prípadov ako ukončovaciu podmienku uvažujeme určitý počet výpočtových cyklov.

Procedúra 2: Horolezecký algoritmus

1: Inicializácia počiatočného riešenia θ_k a inicializácia kroku $k \leftarrow 0$;
2: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia;
3: Náhodný výber nového kandidáta $\theta^c \in \mathcal{N}(\theta_k)$ zo susedov.
4: Akceptovanie nového kandidáta $\theta_{k+1} \leftarrow \theta^c$ v prípade, že $\mathcal{F}(\theta^c) > \mathcal{F}(\theta_k)$.
5: Ďalší krok $k \leftarrow k + 1$.
6: Kým nie je splnená ukončovacia podmienka, skočiť na bod 2.

V literatúre uvádzajú aj mierne modifikácie horolezeckého algoritmu, ako napr. zmenu v 3. bode algoritmu. V tomto prípade namiesto otestovania jediného kandidáta z $\mathcal{N}(\theta_k)$, sa overí celé okolie aktuálneho riešenia, z ktorého v prípade splnenia 4. bodu procedúry postupuje najlepší kandidát. Túto stratégiu nazývajú aj ako "steepest ascent" (greedy prístup), kým verziu uvedenú v Procedúre 2 nazývajú aj ako "first improvement" [52, 79].

Horolezecký algoritmus patrí do skupiny jednoprvkových meta-heuristík so statickou účelovou funkciou bez pamäte.

3.3.2 Tabu search

Tabu Search je ďalšou variantou horolezeckého algoritmu. Jedná sa o rozšírenie HC o krátkodobú pamäť, ktorá slúži na riadenie prehľadávania. Autorom tejto metódy je F. Glover [55]. Základný princíp spočíva v ukladaní navštívených kandidátov pomocou fronty (FIFO) $\mathcal{T}$, nazývaný aj ako "tabu list" . V nasledujúcich iteráciách sú prvky uložené v tejto fronte ignorované, čím sa zabraňuje ich opakovanému testovaniu. Čiže formálne sa susedia riešenia $\mathcal{N}(\theta_k)$ redukujú na $\mathcal{N}(\theta_k) - \mathcal{T}$.
Veľkosť fronty $\mathcal{T}$ môže byť variabilná. V prípade že sa táto fronta zaplní, pridanie nového prvku do fronty spôsobí uvoľnenie (výber) prvku z fronty, ktorý sa tam nachádzala najdlhšie (najviac iterácií).

Procedúra 3: Tabu Search

1: Inicializácia počiatočného riešenia θ_k a inicializácia kroku $k \leftarrow 0, \mathcal{T} \leftarrow \emptyset$;
2: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia;
3: Výber nového kandidáta θ^c pri $\theta^c \in \mathcal{N}(\theta_k) - \mathcal{T}$.
4: Akceptovanie nového kandidáta $\theta_{k+1} \leftarrow \theta^c$ v prípade, že $\mathcal{F}(\theta^c) > \mathcal{F}(\theta_k)$.
5: Ďalší krok $k \leftarrow k + 1$, pridanie θ_k do fronty $\mathcal{T}$ (prípadné odobratie posledného prvku z fronty).
6: Kým nie je splnená ukončovacia podmienka, skočiť na bod 2.

Daná procedúra sa dá rozšíriť o pridanie dlhodobej pamäte, v ktorej sa ukladajú najčastejšie riešenia z predchádzajúcich iterácií. Dlhodobá pamäť sa použije na negatívne ohodnotenie (znevýhodnenie) často používaných kandidátov, ktoré nie sú pri testovaní

zahrnuté do krátkodobej pamäte [52, 68].

Tabu Search patrí do skupiny jednoprvkových meta-heuristík so statickou účelovou funkciou a s pamäťou.

3.3.3 Simulované žíhanie

Simulované žíhanie [55] je meta-heuristika založená na simulovaní fyzikálnych procesov pri žíhaní ocele. Cieľom tohto fyzikálneho procesu je odstránenie defektov ocele (kryštalickej mriežky). Princíp je v zohriatí a pomalom ochladzovaní - žíhaní [52]. Zohriatie spôsobuje zánik defektov, kým pomalé vychladzovanie znižuje pravdepodobnosť vzniku nových defektov. V popise algoritmu sa vo všeobecnosti zachováva pôvodné pomenovania parametrov priamo z fyzikálneho procesu.
Simulované žíhanie je variantou horolezeckého algoritmu (rozšírenie v bode 4). Zásadným rozdielom je možnosť prijatia nového kandidáta θ^c aj v prípade, že $\mathcal{F}(\theta^c) < \mathcal{F}(\theta_k)$ s určitou časovo sa meniacou (klesajúcou) pravdepodobnosťou $P(T_k, \theta^c, \theta_k)$:

$$P(T_k, \theta^c, \theta_k) = e^{\frac{\theta^c - \theta_k}{T_k}} \tag{3.2}$$

Táto metóda je parametrizovaná s premennou T a konštantou δ, kde T vyjadruje teplotu a δ vyjadruje mieru klesania teploty. Parameter T sa mení v každej iterácií podľa $T_{k+1} = T_k\delta$, kde δ sa nachádza v intervale[3] $(0, 1)$. Meniace sa T určuje meniacu sa pravdepodobnosť prijatia zlého kandidáta. Táto pravdepodobnosť je závislá aj od veľkosti rozdielu kvality (energie) kandidáta a aktuálneho riešenia. Je teda zrejmé, že po danom počte iterácií poklesne pravdepodobnosť $P(T_k, \theta^c, \theta_k)$ na takú nízku hodnotu, že sa táto metóda bude správať ako jednoduchý horolezecký algoritmus [15]. Možnosť prijatia horších výsledkov má zabrániť zaseknutiu v lokálnych extrémoch. [55]

Procedúra 4: Simulované žíhanie

1: Inicializácia počiatočného riešenia θ_k a inicializácia kroku $k \leftarrow 0$, T a δ;
2: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia;
3: Výber nového kandidáta $\theta^c \in \mathcal{N}(\theta_k)$ zo susedov.
4: Akceptovanie nového kandidáta $\theta_{k+1} \leftarrow \theta^c$ v prípade, že $\mathcal{F}(\theta^c) > \mathcal{F}(\theta_k)$, alebo keď $\gamma < e^{\frac{\theta^c - \theta_k}{T}}$, kde γ je náhodné číslo z intervalu $\langle 0, 1\rangle$.
5: Ďalší krok $k \leftarrow k + 1$.
6: Úprava teploty $T_{k+1} \leftarrow T_k\delta$.
7: Kým nie je splnená ukončovacia podmienka (väčšinou T_{k+1} blízke 0), skočiť na bod 2.

Základný princíp simulovaného žíhania môžeme adaptovať aj v jednoduchšom kontexte, a to pridaním časovo sa meniacej pravdepodobnosti prijímania horších výsledkov. Na začiatku sa zvolí hodnota δ, ktorá sa v každej iterácií zmenšuje (z δ na 0),

[3] Väčšinou sa δ zvolí ako $0 \ll \delta < 1$ [52].

kým sa nevykoná určený počet iterácií k_δ. Miera prijatia horších výsledkov je potom $p = \delta(1 - \frac{k}{k_\delta})$.

Procedúra 5: Zjednodušené simulované žíhanie

1: Inicializácia počiatočného riešenia θ_k a inicializácia kroku $k \leftarrow 0$ a δ.
2: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia;
3: Výber nového kandidáta θ^c zo susedov θ_k, $\theta^c \in \mathcal{N}(\theta_k)$.
4: Akceptovanie nového kandidáta $\theta_{k+1} \leftarrow \theta^c$ v prípade, že $\mathcal{F}(\theta^c) > \mathcal{F}(\theta_k)$, alebo keď $\gamma < (1 - \frac{k}{k_\delta})\delta$, kde γ je náhodné číslo z intervalu $\langle 0, 1\rangle$.
5: Ďalší krok $k \leftarrow k + 1$.
6: Kým nie je splnená ukončovacia podmienka (napr. $k = k_{max}$), skočiť na bod 2.

Simulované žíhanie (SA) zaraďujeme do skupiny jednoprvkových meta-heuristík so statickou účelovou funkciou bez pamäti (ukladanie hodnôt T a δ sa neberie do úvahy).

3.4 Prírodou inšpirované meta-heuristiky

Posledné roky sa pozornosť výskumu z oblasti meta-heuristík zameriava na prírodou inšpirované metódy. Ich história siaha až do roku 1970, kedy J. Holland z Michigenskej Univerzity prvýkrát popísal *genetické algoritmy* [68, 55]. Väčší rozmach výskumu prírodou inšpirovaných metód nastal ale až v posledných 20 rokoch [112]. Tieto metódy tvoria špeciálnu skupinu meta-heuristík. Ich základný princíp spočíva vo využití procesov odpozorovaných z prírody. Niektoré (ako napr. genetické algoritmy) sú založené na princípe prirodzenej evolúcie [88]. Ďalšiu skupinu tvoria metódy inšpirované správaním rôznych druhov živočíšnej ríše (ako napr. mravce, včely atď.) [112, 111]. Medzi najznámejšie metódy patria[4]: *ant colony optimization* (1992), *particle swarm optimization* (1995), *harmony search* (2001), *honey bee algorithm* (2004), *virtual bee algorithm* (2005), *firefly algorithm* (2007), *cuckoo search* (2009), *eagle strategy* (2010), *bat algorithm* (2010) a *grey wolf optimizer* (2014) [111, 113, 109, 114, 110, 112, 64].
Výhodou prírodou inšpirovaných algoritmov je ich jednoduchosť a flexibilita. Sú veľmi jednoduché na implementovanie v rôznych programovacích jazykoch a sú schopné riešiť široké spektrum optimalizačných úloh [112].
Väčšina prírodou inšpirovaných algoritmov je populačná, čiže pracuje s množinou kandidátov riešenia. Kandidáti sa v tomto prípade nazývajú aj ako *jedinci* alebo *agenti* [112]. Značnú časť prírodou inšpirovaných algoritmov tvorí skupina nazývaná *swarm intelligence* [112], kde sa napodobňuje sociálne správanie kolónií hmyzu. Kvôli podobnosti algoritmov sa však sem zaradujú aj algoritmy inšpirované inými zástupcami živočíšnej ríše ako hmyz, kde sa jedná o spolupracujúcich agentov (ďalej pozri kapitolu 3.4.3).

[4]Prehľad aktuálnych prírodou inšpirovaných meta-heuristík je v [64], ktoré však neuvádzame pre ich rozsiahlosť.

Veľká časť prírodou inšpirovaných meta-heuristík slúži na riešenie spojitých optimalizačných problémov, kde riešenie je reprezentované množinou reálnych čísiel $\theta \in \mathbb{R}^n$ [112]. Metódy v tejto verzii nie sú priamo adaptovateľné pre diskrétne problémy ako napr. permutácia celých čísiel. Aby sa tieto metódy dali použiť pre diskrétny optimalizačný problém, je možné využiť dve možnosti [112, 46]. Prvá možnosť je *reprezentovanie* riešenia z *diskrétneho* priestoru a zmena potrebných častí algoritmu. Napr. pri permutačných problémoch (riešenie je reprezentované permutáciou) na modifikovanie riešenia sa použijú len operácie, ktoré produkujú platnú permutáciu [46]. Druhá možnosť je ponechať metódy v pôvodnom stave a *zmenit' reprezentáciu* hodnôt zo spojitého priestoru na diskrétny pri hodnotení účelovou funkciou. Takáto zmena[5] reprezentácie riešenia prebieha pomocou rôznych kódovaní [112].

V nasledujúcej časti popisujeme niektoré vybraté prírodou inšpirované meta-heuristiky.

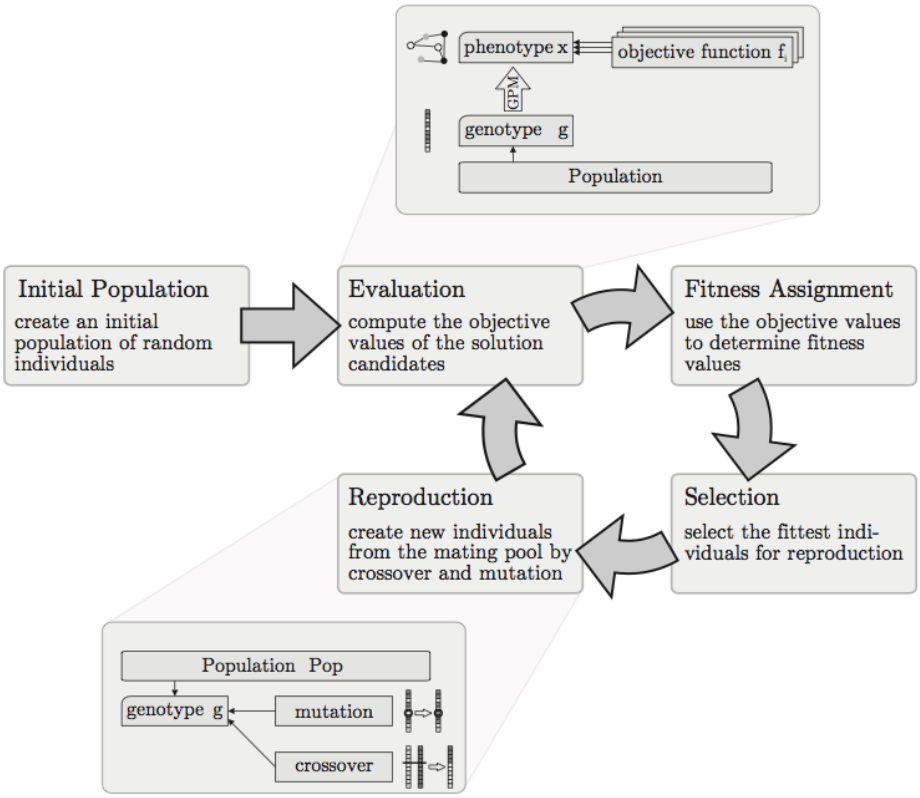
3.4.1 Genetické algoritmy

Genetické algoritmy sú založené na princípe prirodzenej evolúcie podľa Charlesa Darwina. Základnými stavebnými prvkami týchto metód sú *gény*. Z génov pozostávajú *chromozómy* $\theta \in \Theta$ (nazývané aj reťazce alebo jedinci). *Populáciou* $\mathtt{P} = \{\theta_{k_1}, \ldots, \theta_{k_r}\}$ sa označuje aktuálny "stav" algoritmu, pozostávajúci z množiny jedincov. Meniaca sa populácia v príslušnej iterácii algoritmu sa nazýva *generácia*[6]. [88]
Populácia $\mathtt{P}_k$ sa mení a vyvíja pomocou niekoľkých genetických operátorov ako mutácia a kríženie. Pod pojmom *mutácia* sa rozumie taká operácia, pri ktorej náhodne zvolený gén alebo viacero génov sa zmenia na inú náhodnú hodnotu. Mutácia vnáša novú informáciu do chromozómov. *Kríženie* je taká operácia, kde si dvaja jedinci navzájom vymenia niekoľko génov. Pri konštrukcií novej populácie ($\mathtt{P}_{k+1}$) okrem spomenutých genetických operátorov hrá dôležitú úlohu procedúra nazývaná ako *výber*[7] jedincov. Z aktuálnej generácie sa na základe zvolenej stratégie vyberajú niektorí jedinci, na ktorých sa potom aplikujú operácie kríženia a/alebo mutácie. Pri GA je zvykom posunúť najlepšie riešenie danej populácie do nasledujúcej generácie bez akýchkoľvek zmien. Tento postup sa nazýva *elitarizmus*. [88, 52]
Spôsob definovania stratégie výberov, veľkosti populácie ako aj genetických operácií kríženia a mutácie ovplyvňujú efektivitu genetických algoritmov. Operácia mutácie má za úlohu sprostredkovať malé zmeny a tým zaručiť rýchlejšiu konvergenciu k optimu. Zdrojom veľkých zmien je operácia kríženia. Voľba stratégie výberu ako aj genetické operácie ovplyvňujú diverzitu a selekčný tlak metódy [89].
Selekčný tlak vyjadruje mieru uprednostnenia najlepších kandidátov oproti ostatným [89]. Nastavenie vysokej hodnoty selekčného tlaku spôsobuje rýchlejšiu konvergenciu k najbližšiemu lokálnemu extrému [89]. Diverzita na druhej strane vyjadruje rôznorodosť, ktorá vnáša nové vlastnosti do kandidáta [89, 112].

[5] Rôzne možnosti takýchto transformácií sú uvedené v [112].
[6] Generácia je populácia v danom výpočtovom cykle procedúry.
[7] Najpoužívanejšie výbery sú: turnajový, ruletový, náhodný. Bližší popis rôznych výberov je uvedený napr. v [88] a v [52].

V prípade riešenia problému s veľkým počtom lokálnych extrémov je vhodné si zachovať diverzitu populácie, aby sa algoritmus nezasekol v lokálnych extrémoch. Dôležitú vlastnosť GA predstavuje schopnosť opustiť lokálny extrém a následne pokračovať v prehľadávaní priestoru riešení a smerovať ku globálnemu optimu. [89]

Základný princíp fungovania genetických algoritmov je znázornený na obrázku 3.1. Priebeh genetického algoritmu môžeme všeobecne popísať pomocou Algoritmu 6. Vy-



Obr. 3.1: Princíp fungovania genetických algoritmov [106]

tváranie novej generácie (nazývaná aj ako proces reprodukcie [52]) prebieha v bodoch 3 až 9. Najprv sa vyberie subpopulácia z aktuálnej generácie pomocou zvoleného výberu, na ktorú sa aplikujú rôzne genetické operácie ako kríženie a mutácia. Tieto subpopulácie vytvárajú novú populáciu do ďalšej generácie.

Procedúra 6: Genetický algoritmus

1: Inicializácia počiatočnej populácie $k \leftarrow 0$, $\mathtt{P}_k \leftarrow \{\theta_{k_1}, \ldots, \theta_{k_n}\}$.
2: Ohodnotenie celej populácie $\mathtt{P}_k$ pomocou fitness funkcie $\mathcal{F}(\theta)$.
3: **for** $j \leftarrow 1$ **to** r **do**
4: Výber subpopulácie na základe zvoleného výberu $\mathtt{V}_j$, $\mathtt{P}_{sub_j} \leftarrow \mathtt{V}_j(\mathtt{P}_k), \mathtt{P}_{sub_j} \subset \mathtt{P}_i$.
5: Voľba genetického operátora.
6: Nájdenie susedov $\mathcal{N}(\mathtt{P}_{sub_j})$ - aplikovanie genetického operátora.
7: Výber nového kandidáta $\theta_j^c \in \mathcal{N}(\mathtt{P}_{sub_j})$.
8: $j \leftarrow j + 1$.
9: **end for**
10: $\mathtt{P}_{k+1} \leftarrow \theta_1^c \cup \theta_2^c \cup \ldots \cup \theta_r^c$
11: Vstup do d'alšej generácie, $k \leftarrow k + 1$.
12: Kým nie je splnená ukončovacia podmienka, skočiť na bod 2.

V prípade genetických algoritmov je riešenie diskrétnych optimalizačných ako permutačných úloh jednoduché. Genetické operátory kríženia a mutácie sú definované ako zmena permutácie. Malá zmena - mutácia - je vykonaná výmenou dvoch prvkov v permutácií. Kríženie môžeme definovať ako kombináciu dvoch permutácií, bližšie popísané v [88, 52].

Genetické algoritmy patria do skupiny populačných, prírodou inšpirovaných metaheuristík so statickou účelovou funkciou.

3.4.2 Paralelné genetické algoritmy

Paralelné genetické algoritmy (PGA) predstavujú vyššiu úroveň organizácie genetických algoritmov. Základná myšlienka je distribúcia výpočtu do viacerých nezávislých častí, ktoré navzájom vymieňajú informácie a tým kooperujú [89].
Výpočet prebieha na nezávislých a navzájom izolovaných subpopuláciách, ktoré nazývame *ostrovy*[8] [89]. Každý ostrov predstavuje konkrétnu inštanciu GA. Výmena informácie medzi ostrovmi prebieha pomocou *migrácie* [89]. V prípade migrácie je potrebné určiť čas a spôsob migrácie, tzv. *migračný model*. Počas migrácie sa zoberie jeden alebo viacero jedincov z jedného ostrova a nahradia rovnaký počet jedincov v druhom ostrove. Najčastejšie použité [89] stratégie výberu jedincov, ktoré postupujú do migračného procesu z ostrova A na ostrov B je výber najlepších/náhodných z A a nahradenie náhodných/najhorších v B. To, že medzi ktorými ostrovmi prebieha migrácia určuje *topológia* migrácie. Topológia predstavuje architektúru migračných väzieb [89].

[8] Zvyčajne sa používa 4 a viac ostrovov [89].

Procedúra 7: Paralelný genetický algoritmus

1: Inicializácia ostrovov $O = \{O_1 \dots O_n\}$, migračného modelu M, čas migrácie $l_{migrate}$ a topológie T.
2: Inicializácia každého ostrova, $k \leftarrow 0$, $l \leftarrow 0$.
3: Nezávislý beh, 1 výpočtový cyklus každého ostrova O_i.
4: Ďalší krok $k \leftarrow k + 1$, $l \leftarrow l + 1$.
5: Kým nie je splnená podmienka migrácie $l \leftarrow l_{migrate}$, skočiť na bod 3.
6: Migrácia na základe M a T, $l = 0$.
7: Kým nie je splnená ukončovacia podmienka, skočiť na bod 3.

3.4.3 Swarm intelligence

Swarm intelligence popisuje vlastnosť takého systému, kde sa jedná o decentralizované správanie jednoduchých agentov, ktoré komunikujú medzi sebou a sú ovplyvnené svojím prostredím, v ktorom decentralizácia vedie ku globálnej vlastnosti agentov objavovať [112].

V tejto časti práce uvádzame vybrané prírodou inšpirované algoritmy, ktoré spadajú do horeuvedenej kategórie.

Particle swarm optimisation

Particle swarm optimisation (PSO) je metóda založená na sociálnom správaní skupiny zvierat (agentov). Každý agent reprezentuje potenciálne riešenie daného problému a prehľadáva priestor riešení. Prehľadávanie je ovplyvnené skúsenosťou samotného agenta, ako aj celej skupiny (populácie). Agent má tri parametre: *pozíciu* v n-rozmernom priestore $\theta = (x_1 \dots x_n)$, *rýchlosť* $V = (v_1 \dots v_n)$ a hodnotu *fitness*. Pozícia vyjadruje samotné riešenie (umiestnenie v mnohorozmernom priestore riešení), rýchlosť vyjadruje veľkosť zmeny pozície a fitness vyjadruje kvalitu riešenia. Každý agent si pamätá svoje lokálne najlepšie riešenie l a najlepšie riešenie susedných agentov g. Vo väčšine prípadov sa však kvôli jednoduchosti za susedných agentov považujú všetci agenti. Prehľadávanie priestoru spočíva v nasmerovaní pozície každého agenta (zmenou V) na základe lokálneho a globálneho najlepšieho riešenia. [112, 38]
Správanie agenta prebieha na základe troch pravidiel [14, 112] (vzťah (3.4)):

- každý agent ide vlastnou cestou - nasleduje svoj vybraný smer;
- každý agent mení svoj smer podľa smeru susedov - nasleduje najlepšieho suseda;
- každý agent smeruje k svojej najlepšej predchádzajúcej pozícii.

Tieto pravidlá ovplyvňujú sociálne správanie agentov, t.j. na koľko veria susedným agentom, vlastným skúsenostiam alebo ich aktuálnemu smeru [14]. Susedných agentov je možné špecifikovať rôznymi spôsobmi [14]:

- fyzický sused - ráta sa na základe zvolenej metriky v prehľadávacom priestore, kvôli výpočtovej náročnosti sa málokedy aplikuje;
- sociálny sused - počas inicializácie sa napevno nastaví skupina susedných agentov, ktorá sa počas výpočtových cyklov nemení;
- globálny sused - na redukciu výpočtovej náročnosti je možné v každom výpočtovom cykle ukladať jedno globálne najlepšie riešenie, ktoré sa považuje za susedné riešenie každého agenta.

Priebeh PSO popisuje Procedúra 8 s jedným najlepším globálnym riešením θ_g.

Procedúra 8: Particle swarm optimisation

1: Inicializácia počiatočnej populácie $k = 0$, $\mathtt{P}_k \leftarrow \{\theta_{k_1}, \ldots, \theta_{k_n}\}$;
inicializácia $V_i \in \langle 0, V_{max} \rangle$;
inicializácia lokálnych najlepších riešení $L \leftarrow \mathtt{P}_k$, $L = \{l_1, \ldots, l_n\}$.
2: **for all** $\theta_i \in \mathtt{P}_k$ **do**
3: **if** $\mathcal{F}(\theta_i) > \mathcal{F}(l_i)$ **then**
4: $l_i \leftarrow \theta_i$
5: **end if**
6: **end for**
7: $\theta_g \leftarrow arg\max(L)$
8: **for all** $\theta_j \in \mathtt{P}_k$ **do**
9: Zmeň V_j na základe θ_j, l_j a θ_g (vzťah (3.4)).
10: $\theta_j^c \leftarrow \theta_j + V_j$
11: **end for**
12: $\mathtt{P}_{k+1} \leftarrow \bigcup_{i=1}^{n} \theta_i^c$
13: Vstup do ďalšej generácie, $k \leftarrow k + 1$.
14: Kým nie je splnená ukončovacia podmienka, skočiť na bod 2.

Inicializácia vygeneruje skupinu náhodných riešení, čiže náhodné pozície θ a náhodné rýchlosti V, kde r je náhodné číslo, $0 \leq r \leq 1$.

$$\theta_i = \theta_{min} + r(\theta_{max} - \theta_{min}) \tag{3.3}$$

Novú rýchlosť agenta s pozíciou θ_i, aktuálnou rýchlosťou V_i , lokálnou maximálnou pozíciou L_i a globálnou maximálnou pozíciou $\theta_g = G$ rátame ako [112]:

$$V_i = wV_i + c_1\gamma_1(L_i - \theta_i) + c_2\gamma_2(G - \theta_i). \tag{3.4}$$

Následne sa zmení pozícia agenta podľa novej rýchlosti ako:

$$\theta_i = \theta_i + V_i. \tag{3.5}$$

Rýchlosť agentov sa teda mení na základe starej rýchlosti, lokálneho najlepšieho riešenia a globálneho najlepšieho riešenia, kde w, c_1, c_2 sú konštanty (w vyjadruje zotrvačnosť, čiže mieru prehľadávania algoritmu; c_1, c_2 vyjadrujú mieru prispôsobenia sa

lokálnym a globálnym hodnotám) a γ_1, γ_2 sú náhodné čísla z intervalu $\langle 0, 1\rangle$.

V prípade riešenia diskrétnych úloh, môžeme použiť transformácie spojitého priestoru na diskrétny, alebo priamo zmeniť reprezentáciu základných operácií (častí) algoritmu podľa zadaného optimalizačného problému [38, 112]. Rôzne možnosti vytvorenia diskrétnej verzie PSO (ďalej len DPSO) popisujú v [14, 37] a v [38].

Firefly algoritmus

Táto metóda je založená na správaní svetlušiek na základe bioluminesencie [112, 111, 109]. Základom je vzájomné ovplyvnenie (zmena) prvkov populácie kandidátov θ_i (svetlušky) podľa ich kvality $\mathcal{I}_i$ (intenzita bioluminesencie svetlušky). Kvality agentov ovplyvňujú smer prehľadávania priestoru riešení tak, že menej hodnotené riešenia (menšia intenzita bioluminesencie) začnú meniť svoju pozíciu v smere lepšie hodnotených (väčšia intenzita bioluminesencie), čiže prehľadávanie je nasmerované do oblasti, kde sa nachádzajú kvalitnejšie riešenia. Pozícia predstavuje umiestnenie v n-rozmernom priestore $\theta = (x_1, \ldots, x_n)$. V prípade, že sa v blízkosti aktuálneho riešenia nenachádza lepšie riešenie, aplikuje sa náhodná zmena kandidáta (svetluška sa posunie náhodným smerom). Intenzita sa pri inicializácií algoritmu vyráta na základe fitness funkcie $\mathcal{F}(\theta)$. Poloha agentov sa upravuje podľa atraktivity β ostatných svetlušiek, kde atraktivita je pomer intenzity svetla a vzdialenosti od ďalšej svetlušky. Atraktivita kandidáta[9] je $\beta(r_{ij}, \gamma, \beta_0) = \beta_0 e^{-\gamma r_{ij}^2}$, kde β_0 je atraktivita pre vzdialenosť $r = 0$ a γ je miera absorpcie svetla. Poloha kandidáta podľa vzdialenosti r a atraktivity β sa upravuje[10] ako [111]:

$$\theta_i = \theta_i + \beta(\theta_j - \theta_i) + \alpha\epsilon_i \tag{3.6}$$

Vzdialenosť svetlušiek i a j na pozíciách $\theta_i = (x_{i,1}, \ldots, x_{i,d})$ a $\theta_j = (x_{j,1}, \ldots, x_{j,d})$ sa ráta pomocou euklidovskej vzdialenosti:

$$r_{ij} = \sqrt{\sum_{k=0}^{d} (x_{i,k} - x_{j,k})^2} \tag{3.7}$$

Novú polohu svetlušky i ovplyvnenú atraktívnejšou svetluškou j, na základe ich vzdialenosti r_{ij} s mierou absorbcie svetla γ (a $\beta_0 = 1$) môžeme vyjadriť pomocou vzťahu (3.8). Prvé dve zložky ovplyvňujú polohu na základe vzdialenosti svetlušky od atraktívnejšej; tretia zložka vnáša náhodnú zmenu pomocou vektoru náhodných čísiel ϵ [109].

$$x_i^c(\theta_i, \theta_j, r_{ij}, \gamma, \epsilon) = (1 - \frac{1}{1 + \gamma r_{ij}^2})\theta_i + \frac{1}{1 + \gamma r_{ij}^2}\theta_j + \alpha\epsilon_i \tag{3.8}$$

[9] Na zrýchlenie výpočtov sa atraktivita vo väčšine prípadov aproximuje ako $\beta(r_{ij}, \gamma, \beta_0) = \frac{\beta_0}{1+\gamma r_{ij}^2}$ [111].

[10] V prípade väčšiny problémov sa uvažuje $\beta_0 = 1$, $\epsilon_i \in (rand() - \frac{1}{2})^n$ a $0 \leq \alpha \leq 1$ [111].

Priebeh Firefly (d'alej len FFA) popisuje Procedúra 9.

Procedúra 9: Firefly algoritmus

1: Inicializácia počiatočnej populácie $k \leftarrow 0$, $\mathtt{P}_k \leftarrow \{\theta_1, \ldots, \theta_n\}$, $\theta_{best} \leftarrow \theta_1$.
2: Vyjadrenie intenzity svetla populácie $\mathtt{P}_k$ pomocou fitness funkcie, $I_i \leftarrow \mathcal{F}(\theta_i)$, nastavenie mieru absorpcie svetla γ.
3: **for all** $\theta_i \in \mathtt{P}_k$ **do**
4: **for all** $\theta_j \in \mathtt{P}_k$ **do**
5: Pozn.: intenzita I_i je vyjadrená priamo hodnotou $\mathcal{F}(\theta_i)$.
6: **if** $I_i < I_j$ **then**
7: Vygeneruj ϵ_i, vyrátaj r_{ij}.
8: Pozn.: attraktivita sa mení podľa vzdialenosti r_{ij} riešení θ_i a θ_j.
9: $\theta_i^c \leftarrow$ posun θ_i smerom k θ_j na základe r_{ij} podľa vzťahu (3.8).
10: Ohodnotenie, zmena intenzity svetla nového riešenia θ_i^c.
11: Nahradenie θ_i s θ_i^c.
12: **end if**
13: **end for**
14: **end for**
15: Vstup do d'alšej generácie, $k \leftarrow k + 1$.
16: **for all** $\theta_i \in \mathtt{P}_k$ **do**
17: **if** $\mathcal{F}(\theta_{best}) < \mathcal{F}(\theta_i)$ **then**
18: $\theta_{best} \leftarrow \theta_i$.
19: **end if**
20: **end for**
21: Malá náhodná zmena v θ_{best}.
22: Kým nie je splnená ukončovacia podmienka, skočiť na bod 3.

Keď sa z Procedúry 9 odstráni krok 4 a I_j sa nahradí intenzitou najlepšieho riešenia populácie, dostávame priamo PSO [109]. Na podobnom princípe prehľadávania priestoru (s miernymi modifikáciami, iné zvieratá) fungujú aj iné meta-heuristík ako napr. Bat algorithm [110].

Možným vylepšením tejto metódy je modifikácia veľkosti kroku α prehľadávania (vzorec 3.6) na základe informácie niekoľkých predošlých krokoch [115]. Základná myšlienka je zmenšenie kroku prehľadávania v blízkosti optimálnych riešení a zväčšenie kroku v opačnom prípade.

V prípade diskrétneho problému je nutná modifikácia niektorých častí algoritmu tak, ako v prípade PSO. V [46], [111] a [85] uvádzajú ukážku diskrétnej verzie FFA, ktorú je možné aplikovať na permutačný problém.

3.5 Oblasti využitia

Pri riešení optimalizačného problému je otázkou, ktorá optimalizačná metóda je najlepšia pre daný problém. Na základe rôznorodosti optimalizačných problémov a veľkého počtu týchto metód existuje tzv. "no free lunch theorem" . Táto teória hovorí, že neexistuje jedna najlepšia metóda vhodná pre riešenie všetkých tried optimalizačných problémov [90].
Vzhľadom na počet a rôznorodosť meta-heuristických algoritmov, uvedieme len zopár možností aplikácie meta-heuristík. Tieto údaje sú čerpané z [6, 68, 106] (názvy problémov sú v anglickom jazyku): Quadratic Assignment Problem, Combinatorial Optimisation, MAXSAT, Graph planarization, Steiner tree problem, TSP, Image Processing, Simulation Optimization problem, Vehicle Routing, Job Shop Scheduling, Linear Ordering Problem, Resource - Constrained Project Scheduling, Data mining.

Meta-heuristicky našli svoje uplatnenie aj pri lúštení klasických šifier. Najviac pozornosti sa venovalo lúšteniu monoalfabetickej substitúcie a transpozičných šifier.
V nasledujúcej časti uvedieme[11] niekoľko autorov a ich dosiahnuté výsledky pri lúštení klasických šifier pomocou meta-heuristík.

3.5.1 Lúštenie klasických šifier pomocou meta-heuristík

Spillman a kol. v článku [96] použili GA na lúštenie jednoduchej substitúcie. Priestor riešení definovali ako všetky možné permutácie kľúča. Účelová funkcia bola skonštruovaná na základe rozdielu monogramov a bigramov od referenčných hodnôt. Táto funkcia je ďalej rozšírená s delením a umocnením konštantou na "zmiernenie" malých a veľkých rozdielov pri porovnaní, bez bližšieho vysvetlenia. Pomocou účelovej funkcie hodnotili priamo text dešifrovaný pomocou kľúča (kandidáta na riešenie). Zaoberali sa možnosťami definovania genetických operátorov ako mutácia a kríženie v prípade substitučnej šifry. Taktiež analyzujú vplyv pravdepodobnosti mutácie na úspešnosť. Zistili, že zväčšovaním pravdepodobnosti mutácie algoritmus stráca svoju efektivitu. Najlepšie výsledky dosiahli pri pravdepodobnosti mutácie 0.05, kde dosiahli úspešnosť v priemere okolo 90% zhody kandidáta so správnym riešením. V experimentoch použili fixnú veľkosť populácie 10 prvkov, mutáciu definovali ako výmenu dvoch prvkov v kľúči a na kríženie použili špeciálnu metódu založenú na vzájomnej výmene znakov v dvoch kľúčoch založenú na veľkosti výskytu znakov.
Na záver popisujú závislosť efektivity GA od nastavených parametroch.

Matthews v [61] použil GA na lúštenie transpozičných šifier (jednoduchú tabuľkovú transpozíciu). Pomocou GA hľadá správnu veľkosť bloku transpozície a následne správnu permutáciu. Účelovú funkciu skonštruoval na základe hodnotenia výskytu najčastejších zvolených bigramov a trigramov. Účelová funkcia taktiež obsahovala ne-

[11] Uvedené publikácie sú zoradené na základe roku publikácie. Od 1993 do 2016.

gatívne hodnotenie na základe výskytu nemožných trigramov. Úplne dešifrovať šifru dokázali len zriedkavo. Na záver autor odkazuje na možnosť zvýšenia efektivity GA pomocou zmeny parametrov algoritmu.

Forsyth a Safavi-Naini v [20] použili SA na lúštenie jednoduchej substitúcie. Priestor riešení definovali ako všetky možné permutácie kľúča. Účelová funkcia bola skonštruovaná na základe rozdielu bigramov od referenčných hodnôt. Testovali úspešnosť pri textoch rôznych žánrov v závislosti od dĺžky textov. Dosiahli úspešnosť až 100% zhody kandidáta so správnym riešením pri dĺžkach textov 4000 a 5000 znakov. V prípade textov dĺžky 2000 znakov dosahovali variabilnú úspešnosť medzi 0 a 100% zhody so správnym riešením. V prípade textov dĺžky 1000 znakov mali pri väčšine žánrov 0% zhodu.

Jakobsen [41] uviedol veľmi elegantný spôsob optimalizácie pri lúštení jednoduchej substitúcie. Podobne ako [20], prehľadáva priestor kľúčov. Pri riešení použil HC (aj keď to priamo v článku takto nenazval), kde v kľúči jednoduchej substitúcie vymieňal dvojice znakov počas prehľadávania. Na hodnotenie použil Manhattanskú vzdialenosť bigramov od referenčných hodnôt. Hlavným prínosom jeho práce je odstránenie dešifrovacieho procesu pri hodnotení kľúča. Myšlienka spočíva v tom, že pri zmene dvojíc v kľúči (permutácií) nie je nutné opakovanie dešifrovať text pre zmenený kľúč a následne zrátať štatistiku bigramov. Táto štatistika sa zachováva okrem dvoch riadkov a stĺpcov ktoré boli vymenené v kľúči a tým pádom stačí túto štatistiku vyrátať pre prvý kľúč pri prehľadávaní. Následne pri zmene v kľúči sa vymenia príslušné hodnoty v matici, kde je uložená príslušná štatistika. V článku ďalej uvádza možnosti inicializácie pomocou frekvencie grafém.

Bagnall a kol. [99] použili GA na prehľadávanie priestoru kľúča 3 resp. 4 rotorového (všeobecného modelu) šifrátora. Úplne však nedokázali určiť správne nastavenie všetkých rotorov.

Jedným z prelomových prác v tejto oblasti je dizertačná práca A. Clarka [13]. Clark porovnáva úspešnosť TS, GA a SA pri lúštení jednoduchej substitúcie a transpozičných šifier. Hlavným prínosom tejto práce oproti starším publikáciám iných autorov je porovnanie úspešnosti lúštenia na základe rôznych účelových funkcií. Na konštrukciu účelovej funkcie použil a kombinoval štatistiky n-gramov pri rôznom váhovaní. Preukázal možnosť získania dobrých výsledkov pri použití viacerých kritérií a pri správnom váhovaní. Porovnal úspešnosť týchto algoritmov na základe dĺžky textu, podľa veľkosti prehľadaného priestoru a podľa potrebného času výpočtov. Najvhodnejšou metódou na lúštenie jednoduchej substitúcie a transpozičných šifier uvádza TS a účelovú funkciu skonštruovanú na základe trigramov. V prípade lúštenia polyalfabetickej substitúcie dosiahol najlepšie výsledky pomocou použitia paralelných GA.

Russel a kol. [83] použili ACO na automatické lúštenie transpozičných šifier. Jedná sa o špecifickú anagramovú metódu. Na vyhodnotenie kandidátov použili slovník a štatis-

tiku bigramov. Táto metóda úspešne funguje až do veľkosti priestoru riešenia 40!.

Youssef a kol. použili ACO [102] a PSO [103] na lúštenie jednoduchej substitúcie. Na ohodnotenie použili štatistiku monogramov a bigramov. Úspešnosť implementovaných porovnávali pre rôzne dĺžky textov. Podľa uvedených výsledkov dosiahli použiteľné výsledky pre texty dlhšie ako 300 resp. 400 znakov pomocou štatistiky bigramov. Uvedené články sú však bez bližšieho popisu implementácie a bez hlbšej analýzy samotnej problematiky. V [103] použili "diskretizáciu" PSO na základe [38].

M. J. Cowan v [15] uvádza spôsob lúštenia krátkych Playfair šifier pomocou SA. Na ohodnotenie textu používa 4-gramy, čo však detailnejšie nepopisuje.

M.J. Banks [4] vo svojej diplomovej práci pod vedením A. Clarka rozširuje prácu [13]. Popisuje niekoľko dôležitých faktov, ktoré treba mať na zreteli pri lúštení pomocou optimalizačných algoritmov ako výber správnej účelovej funkcie alebo správne nastavenie parametrov optimalizačných algoritmov. Ako hodnotiacu funkciu používa funkciu z [13] s pridaním slovníkového ohodnotenia. Vo svojej práci analyzuje do akej miery ovplyvňujú jednotlivé váhy úspešnosť lúštenia. Ako výsledok uvádza váhy na bigram(0.1), trigram(0.4) a slovník(0.25), kde monogram uvádza ako zbytočnú zložku. Lúštenie bolo zamerané na lúštenie jednoduchej substitúcie pomocou GA.

D. Oranchak [70] vytvoril špecifický útok pomocou GA na známu homofónnu Z408 šifru. Jeho útok spočíval v doplňovaní možných slov na vybratý úsek zašifrovaného textu. Útočil na časť textu (52 znakov - 13%), kde sa nachádzajú najfrekventovanejšie písmená, pomocou ktorých pokrýva až 90% z celkového zašifrovaného textu. Táto metóda sa uvádza ako prvý plne automatický spôsob lúštenia šifry Z408. Napriek úspešnosti tejto metódy treba dodať, že jeho riešenie je prispôsobené na konkrétnu šifru.

R. Hilton [36] použil GA na lúštenie jednoduchej substitúcie. Na ohodnotenie použili váhovanú podobnosť (oproti doteraz použitým vzdialenostiam) frekvencií monogramov, bigramov a trigramov. V prípade monogramov nastavil malú váhu 0.2 kvôli začiatku lúštenia, kde ešte väčšie n-gramy nefungujú dobre podľa jeho zistení. Pre 2-gramy nastavil 0.3 a pre 3-gramy nastavil 0.5. Hlavným prínosom jeho práce je analýza nastavenia vybratých parametrov GA ako veľkosť populácie a veľkosti výberu. Treba však pripomenúť, že jeho implementácia GA je zjednodušená, napr. nepoužíva kríženie a používa len výber najlepších jedincov, na ktoré aplikuje mutáciu. Z jeho výsledkov vyplýva, že veľkosť populácie značne ovplyvňuje úspešnosť, kde pre veľkosť výberu nie je možné odvodiť jednoznačné odporúčanie. Pri porovnaní úspešnosti GA podľa dĺžky textu dosiahol dobré výsledky pre texty nad 1000 znakov.

Singh a kol. v [95] uvádzajú spôsob lúštenia jednoduchej substitúcie pomocou prírodou inšpirovaných algoritmov, konkrétne pomocou FFA. Na ohodnotenie použili funkciu od [96]. Avšak nedosiahli žiadne použiteľné výsledky. Dosiahnuté výsledky porovnávajú s náhodným prehľadávaním. Z článku chýba hlbšia analýza ako použiť FFA pre uvedený

diskrétny problém.

Dhavare a kol. [17] uvádzajú prvý serióznejší pokus lúštenia homofónnej substitúcie. Na lúštenie použili 2 rôzne vrstvy HC. Vnútorná vrstva lúšti monoalfabetickú substitúciu sofistikovaným a rýchlim optimalizovaným spôsobom. Kde vonkajšia vrstva je zodpovedná za transformáciu homofónnej substitúcie na monoalfabetickú (určí sa fixný počet homofónov pre jednotlivé písmena použitej abecedy). Principiálne sa jedná o použitie hyper-heuristiky (aj keď to v článku takto priamo nenazvali). Vonkajšia vrstva hľadá správnu štruktúru kľúča a vnútorná vrstva lúšti na základe zvolenej štruktúry. Vonkajšia vrstva je riadená dosiahnutým hodnotením vnútornej vrstvy. Na hodnotenie použili Manhattanovskú vzdialenosť bigramov od referenčných hodnôt, pričom aplikovali Jakobsenovu [41] optimalizáciu lúštenia. Pre dlhšie texty (nad 3000/4000 znakov) dosiahli veľmi dobré výsledky. Z výsledkov však vyplýva, že ani táto metóda nie je univerzálna a nefunguje dobre pre kratšie texty len v špecifických prípadoch za špecifických predpokladov.

Vobbilisetty a kol. v [105] uvádzajú nový spôsob lúštenia substitučných šifier pomocou Hidden Markov Modelu a HC. Navrhnuté riešenie porovnávajú s výsledkami Jakobsena z [41]. Ako hodnotiacu funkciu používajú Manhattanovskú vzdialenosť bigramov, podobne ako v [41]. Na vylepšenie úspešnosti navrhujú použite veľkého množstva reinicializácií. V prípade 10^5 reinicializácií dosahujú v priemere 74%-nú zhodu so správnym riešením pri textoch dĺžky 200 znakov a 95%-nú zhodu od dĺžky 300 znakov. V prípade menšieho počtu reštartov dosahujú dobré výsledky len v prípade dlhších textov. Z ich výsledkov tiež vyplýva, že použitie Hidden Markov Modelu bez reinicializácie značne zaostáva v porovnaní s výsledkami z [41].

Vo všetkých uvedených prácach chýba presný a systematický popis ako správne použiť meta-heuristiky (a účelové funkcie) v prípade kryptoanalýzy klasických šifier. Hlbšou analýzou jednotlivých častí lúštenia sa autori nezaoberajú.

Kapitola 4

Nové možnosti efektívneho využitia meta-heuristík pri lúštení monoalfabetickej substitúcie

Táto kapitola je venovaná detailnej analýze lúštenia klasických šifier pomocou meta-heuristík. Cieľom tejto časti práce je uviesť ucelený pohľad a odporúčania na dosiahnutie čo najlepších[1] výsledkov pri ich lúštení. Zvolená metodika je demonštrovaná experimentálnym lúštením monoalfabetickej substitúcie.

V prvej časti kapitoly uvádzame základný postup transformácie lúštenia klasických šifier na optimalizačný problém. V ďalších častiach porovnávame rôzne možnosti konštrukcie účelových funkcií s využitím analýzy textu. Prácu autorov z tejto oblasti rozširujeme novými spôsobmi porovnania kvantitatívnych charakteristík textu. Ďalej sa venujeme analýze využitia Markovských modelov rôznych rádov, ako aj možnosti využitia slovníkového ohodnotenia. Následne stanovíme potrebné kritériá, ktoré majú účelové funkcie spĺňať - sekcia 4.1. Experimentálne overenie kvality a porovnanie rôznych účelových funkcií uvádzame v sekcii 4.2. V sekcii 4.3 sa zaoberáme analýzou lúštenia problematických (krátkych) textov.

4.1 Transformácia kryptoanalýzy klasických šifier na optimalizačný problém

Novým prístupom pri *kryptoanalýze* klasických šifier je prehľadávanie priestoru kľúčov daného kryptosystému pomocou meta-heuristík (sekcia 3). Základným krokom tohto procesu je transformácia prehľadávania priestoru kľúčov na optimalizačný problém. K tomu je potrebné, aby kryptoanalýza a prvky kryptosystému (podľa Def. 1) boli prevedené na prvky optimalizačného problému (sekcia 3.1), ktorého najdôležitejšie časti sú

[1] Pričom za úspešné lúštenie považujeme dosiahnutie minimálne 80%-nej zhody grafém nášho výsledku so správnym riešením.

reprezentácie kandidáta na riešenie θ a účelová funkcia $\mathcal{F}$. Po transformácií základných prvkov nasleduje voľba meta-heuristiky, ktorou chceme daný optimalizačný problém riešiť. Následne je možné stanovenie počiatočného riešenia θ_0, susedných riešení[2] $\mathcal{N}(\theta)$ [40] a ukončovacej podmienky.

Cieľom kryptoanalýzy je nájdenie tajného kľúča k z $\mathcal{K}$. Kandidátom riešenia θ^c je teda kľúč, alebo množina kľúčov z $\mathcal{K}$. Priestor riešení Θ ($\mathcal{K}$) závisí od známeho šifrovacieho algoritmu. Pri kryptoanalýze vychádzame z niekoľkých faktov:

- Vstupom je známy zašifrovaný text $Y \in \mathcal{C}$.
- Šifrovací a dešifrovací algoritmus $e_k \in \mathcal{E}$ resp. $d_k \in \mathcal{D}$ sú tiež známe.

Dôležitou časťou je vyhodnotenie úspešnosti kandidátov pomocou účelovej funkcie $\mathcal{F}(\theta^c)$. Keďže samotný kľúč nie je možné priamo ohodnotiť, vhodnosť sa overí na základe dešifrovaného textu[3] $X^c = d_k(Y, \theta^c)$ [25].

4.1.1 Limity pri voľbe modelu jazyka

Na matematický popis vlastností prirodzených jazykov sa využívajú tzv. modely jazyka (sekcia 2), v ktorých sú obsiahnuté (pre lúštenie) dôležité charakteristiky, tzv. referenčné hodnoty. Nami zvolené účelové funkcie $\mathcal{F}(\theta)$ (sekcia 4.1.2) porovnávajú podobnosť, resp. vzdialenosť zvolených kvantitatívnych charakteristík textu od stanovených referenčných hodnôt jazyka. Od voľby modelu jazyka závisí, aké charakteristiky budeme porovnávať. Zvolený model môže zvýšiť úspešnosť lúštenia, ale prináša aj obmedzenia. Najčastejšie použité modely jazyka pri kryptoanalýze klasických šifier sú Markovské modely [13, 96, 20, 41, 17] rádov r:

- $r = 0$ - porovnanie grafém (model M_1);
- $r = 1$ - porovnanie 2-gramov (model M_2);
- $r = 2$ - porovnanie 3-gramov (model M_3).

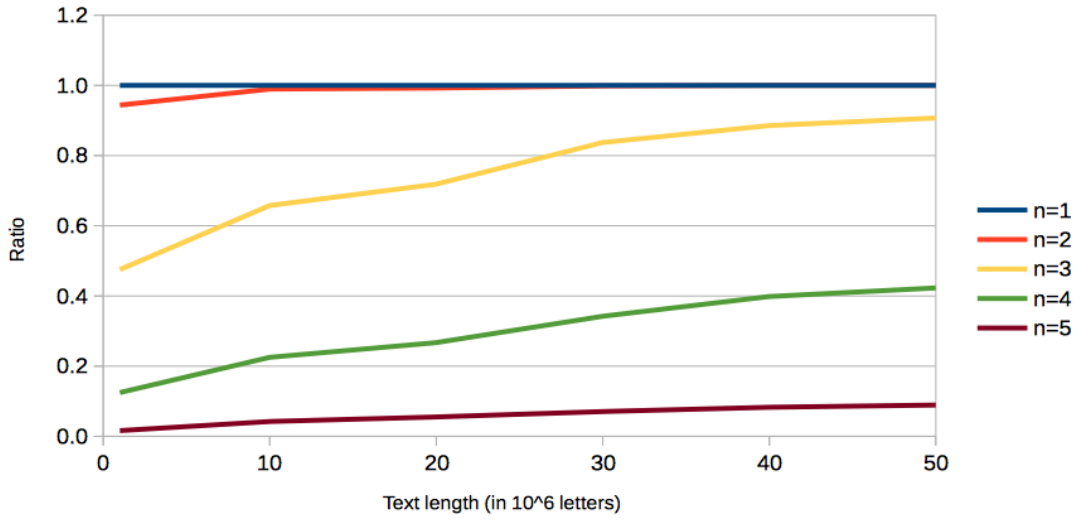
V prípade modelu M_1 sa používa kvantitatívna charakteristika grafém. Z pohľadu presnosti môžeme túto štatistiku považovať za najmenej presnú, keďže nezahŕňa lokálne vzťahy medzi písmenami [29]. V prípade niektorých kryptosystémov, ako napr. transpozičné šifry - kde sa frekvencia písmen nemení po šifrovaní - je tento model nepoužiteľný. Ďalší problém predstavujú málo frekventované grafémy v prípade lúštenia krátkych substitučných šifier.

V prípade modelov vyšších rádov sú vyššie uvedené nedostatky vykompenzované, avšak pri ich voľbe je potrebné mať na zreteli dve dôležité obmedzenia. Prvé sa týka minimálnej dĺžky textu kvôli splneniu požiadavky reprezentatívnosti (sekcia 2.3.2). Druhé

[2] Susedné riešenia $\mathcal{N}$ je možné upresniť až po voľbe konkrétneho kryptosystému.

[3] K vyhodnoteniu kvality X^c je potrebné použiť takú účelovú funkciu $\mathcal{F}$, ktorá je založená na charakteristike prirodzeného jazyka.

obmedzenie sa vzťahuje k časovej zložitosti vyhodnotenia. Faktom je, že v prípade Markovských modelov vyšších rádov M_n existuje veľké množstvo n-gramov (26^n), z ktorých sa však len malá podmnožina reálne vyskytuje v danom jazyku. Podiel výskytu rôznych n-gramov v porovnaní s maximálnym teoretickým počtom rôznych n-gramov (v závislosti od dĺžky korpusu) uvádzame na Obr. 4.1. Ako je z obrázku vidieť, pre $n = 5$ je už tento podiel skoro 0.



Obr. 4.1: Podiel počtu vyskytujúcich sa n-gramov a maximálneho počtu pre rôzne dĺžky OANC korpusu

	M_1	M_2	M_3	M_4	M_5
r	0.01	0.001	0.001	0.0001	0.0001
N	186	210	1579	4511	64237

Tabuľka 4.1: Reprezentatívna dĺžka textu pre rôzne n-gramové modely

V tabuľke 4.1 uvádzame požiadavky na minimálnu dĺžku textu N pre rôzne modely, aby bola splnená požiadavka reprezentatívnosti. Reprezentatívnosť bola vypočítaná pomocou vzorca (2.6) a smerodajné odchýlky r boli určené empiricky.

Z Tabuľky 4.1 vyplýva, že pri lúštení šifrovaných správ z reálneho života, keď dĺžka správy sa väčšinou pohybuje okolo stovky znakov, nie je vhodné priamo použiť modely vyšších rádov, keďže nie je splnená požiadavka reprezentatívnosti. V našich experimentoch sme preto použili modely M_1 a M_2. Modely vyšších rádov však môžeme využiť pri skonštruovaní slovníkov, ktoré môžu pomôcť dodatočne ohodnotiť a vybrať najpravdepodobnejší výsledok z väčšej množiny.

Na základe vyššie uvedeného a údajov ohľadne reprezentatívnej dĺžky (Tabuľka 4.1) sme určili minimálnu dĺžku testovacej vzorky ako *200 znakov*[4].

4.1.2 Špecifikácia účelovej funkcie

Účelové funkcie je potrebné skonštruovať tak, aby ohodnotili text z prirodzeného jazyka. Ohodnotenie textu spočíva v jeho porovnaní so štatistickými (kvantitatívnymi) charakteristikami jazyka [50] na základe stanoveného modelu. Vyhodnotenie úspešnosti kandidátov pomocou $\mathcal{F}(\theta)$ pri lúštení klasických šifier pozostáva z dvoch krokov [25] :

1. Získanie referenčných kvantitatívnych charakteristík jazyka zo základnej vzorky (korpusu)[5].

2. Porovnanie kvantitatívnych charakteristík kandidáta s referenčnými hodnotami[6] zvoleného jazyka.

Porovnanie spočíva vo vyjadrení ich vzdialenosti alebo podobnosti (sekcia 2.3.3). Zvolili sme nasledovnú množinu účelových funkcií[7] $\mathbb{F} = \{\chi^2, CS, L_1, L_2, JSD\}$.

Na ohodnotenie kvality účelových funkcií sme si stanovili nasledovné kritériá:

- Funkcia má priradiť najlepšie skóre správnemu riešeniu.

- Funkcia má vyjadrovať mieru vhodnosti, čo môžeme interpretovať ako percentuálna zhoda grafém a otvoreného textu.

Prvé kritérium sa vzťahuje k základnému predpokladu úspešného využitia meta-heuristík, t.j. dosiahnutie globálneho extrému účelovej funkcie.
Druhé kritérium sa týka priebehu prehľadávania priestoru riešení a umožní zoradenie kandidátov podľa úspešnosti, v našom prípade zmysluplnosti textu. Vytvorenie účelovej funkcie na porovnanie kvality kandidátov $\theta \in \Theta$ je možné na základe dôležitej vlastnosti klasických šifier, ktorá sa nazýva *Utility of partial solution* [36, 25] (neplatí vo všeobecnosti). Táto vlastnosť spočíva v tom, že čiastočne správny kľúč produkuje čiastočne správny text, čo umožňuje aj samotné využitie optimalizačných algoritmov na lúštenie.

[4] Experimenty sme vykonali aj pre správy kratšie ako 200 znakov, ktoré však uvádzame len informatívne.

[5] Tento krok sa nazýva aj ako *off-line* krok.

[6] Tento krok sa nazýva aj ako *on-line* krok.

[7] Vo výsledkoch experimentov sú použité nasledovné pomenovania funkcií: Chi (χ^2 divergencia), Cos (kosínusová podobnosť), Manhattan (Manhattanská vzdialenosť - L_1), Euklid (Euklidovská vzdialenosť - L_2), JSD (Jensen-Shannonova divergencia).

4.2 Lúštenie monoalfabetickej substitúcie

Metodika transformácie kryptoanalýzy na optimalizačný problém uvedená v predchádzajúcej kapitole bude ilustrovaná na monoalfabetickej substitúcie. Pomocou empirických testov budú detailnejšie preskúmané a analyzované možnosti nastavenia účelovej funkcie s čím súvisí aj voľba modelu jazyka a meta-heuristiky[8].

4.2.1 Špecifikácia monoalfabetickej substitúcie ako optimaliačného problému

Monoalfabetickú substitúciu sme špecifikovali podľa F. L. Bauera [5] ako permutáciu na abecede otvoreného textu. Reprezentáciu kryptosystému je možné ďalej upresniť:

- $\mathcal{A}_P = \mathcal{A}_C = \{a, b, \ldots, z\}$.
- $\Theta = \mathcal{K}$ sú všetky permutácie na množine $\mathcal{A}_P$, kde $\theta \in \Theta$ reprezentuje kľúč k.
- Susedné riešenia $\mathcal{N}(\theta)$ sú všetky permutácie získané pomocou výmeny dvoch prvkov v θ, konkrétne $\binom{26}{2}$ permutácií.
- Funkcia $\mathcal{F}$ ohodnocuje text dešifrovaný pomocou k, podľa sekcie 4.1.2.

Ďalšie značenia:

- θ_{corr} - správne riešenie.
- $\mathcal{F}_{max}$ - globálny extrém účelovej funkcie.
- Funkcia $\mathbb{M}(\theta)$ - vyjadruje percentuálnu zhodu grafém θ s θ_{corr}.
- $\mathbb{B}$ - počet $\mathcal{F}(\theta) > \mathcal{F}(\theta_{corr}), \theta \in \mathcal{N}(\theta_{corr})$.

Voľba operácie na zmenu permutácie je dôležitou súčasťou návrhu. Keďže kľúč v tomto prípade predstavuje ľubovolnú permutáciu použitej abecedy, pri použití 26 znakovej abecedy bez ⊔, máme 26! možných permutácií. Výmena dvoch prvkov kľúča spĺňa základnú podmienku - t.j že pomocou aplikácie konečného počtu výmeny dvoch prvkov je možné vytvoriť všetky permutácie [53] - možnosti prehľadania celého priestoru Θ. Táto výmena predstavuje aj najmenšiu možnú zmenu v permutácii.

Keďže úspešnosť metód lúštenia do značnej miery závisí od dĺžky ZT, všetky testy boli realizované na rozličných dĺžkach testovacej vzorky. Ako testovacie vzorky boli použité náhodne vybrané texty dĺžok 50 až 2000 znakov po 50 znakoch. Dĺžky textov boli zvolené zámerne ako krátke, aby viac odrážali v reálnom živote používané krátke šifrované správy. Experimenty sme zopakovali na 100 rôznych textov pre každú zvolenú dĺžku. Referenčné hodnoty kvantitatívnych charakteristík a testovacie vzorky boli získané z

[8] Vo výsledkoch sú použité anglické pomenovania meta-heuristík. Názvom SimulatedAnnealing2 (SA) označujeme zjednodušenú verziu simulovaného žíhania, ktorú sme uviedli v sekcii 3.3.3.

OANC korpusu [2].

Po upresnení reprezentácie bolo potrebné špecifikovať hodnoty unicity distance (viď. sekcia 2.4.2). V prípade monoalfabetickej substitúcie s veľkosťou abecedy 26 znakov, unicity distance U podľa pôvodného vzťahu od Shannona je:

n	1	2	3
$\mathcal{H}_{AJ}$	4.2	3.7	3.3
$\mathcal{R}$	0.5	1	1.4
U	177	89	63

Tabuľka 4.2: Unicity distance monoalfabetickej substitúcie pre rôzne rády modelov n

Keďže hodnoty U v Tabuľke 4.2 nepresahujú minimálnu reprezentatívnu dĺžku textu 200 znakov, nie je potrebné brať ohľad na ne.

4.2.2 Experimentálne overenie kvality účelových funkcií

Kvalitu účelových funkcií budeme hodnotiť na základe dvoch nami stanovených kritérií.

1. Kritériom je, aby funkcia ohodnotila θ_{corr} najlepšou fitness hodnotou, čo závisí od konštrukcie $\mathcal{F}$ ako aj od voľby modelu jazyka $M_i, i = 1, 2, 3$.

2. Kritériom je, aby účelová funkcia správne vyjadrovala $\mathbb{M}(\theta)$. Toto kritérium bude overené aplikáciou meta-heuristiky. Vychádzali sme z predpokladu, že keď meta-heuristika je schopná sa dopracovať k správnemu výsledku, funkcia $\mathcal{F}(\theta)$ vyjadruje $\mathbb{M}(\theta)$ správne.

Poznámka 3 *Časová zložitosť výpočtov závisí na veľkosti $\mathcal{N}(\theta)$. Nesprávna voľba $\mathcal{N}(\theta)$ môže viesť k zlyhaniu, t.j. algoritmus sa neukončí v reálnom čase.*

Priradenie najlepšej fitness hodnoty k správnemu riešeniu

Základný predpoklad vhodnosti $\mathcal{F}$ je, že $\mathcal{F}_{max}$ predstavuje θ_{corr} (zmysluplný, čitateľný text). Splnenie tohto predpokladu ale neznamená, že sa jedná o jednoznačný výsledok. Rovnaká fitness hodnota môže byť totiž priradená k niekoľkým riešeniam, ktoré môžu, ale nemusia byť susediace. V prípade, že množina susediacich riešení $T = \{\theta_0, \theta_1, \ldots \theta_m\}$, má rovnaké ohodnotenie $\forall\theta_i, \theta_j \in T$, $\mathcal{F}(\theta_i) = \mathcal{F}(\theta_j)$, hovoríme o neutrálnom úseku (nazývané aj ako "plateau") [40]. Výskyt neutrálnych úsekov spôsobuje problémy, ktoré treba zohľadniť pri návrhu meta-heuristiky.
V prípade že dosiahnutý extrém nepredstavuje správne riešenie (a nenachádza sa na neutrálnom úseku), je potrebné zistiť, ako "ďaleko" sa nachádza od správneho riešenia. Totiž v prípade, že extrém je blízko k správnemu riešeniu (výsledný text je čiastočne totožný so správnym výsledkom), text môže byť človekom čitateľný.

V rámci experimentu budú otestované všetky $\mathcal{F} \in \mathbb{F}$. Tieto experimenty vykonáme pre modely M_1 a M_2. Napriek obmedzeniam (sekcia 4.1.1) testy vykonáme aj na M_3, tieto výsledky uvádzame len kvôli porovnaniu.
Prvá časť experimentu pozostáva z overenia, či $\mathcal{F}(\theta_{corr}) = \mathcal{F}_{max}$, k čomu sa dopracujeme ohodnotením všetkých $\binom{26}{2}$ susediacich riešení z $\mathcal{N}(\theta_{corr})$. Postup popisujeme v Algoritme 10.

Algoritmus 10: Hľadanie počtu lepšie ohodnotených susedov

Input: Správne riešenie θ_{corr},
účelová funkcia $\mathcal{F}$
Output: $\mathbb{B}$ - počet $\mathcal{F}(\theta) > \mathcal{F}(\theta_{corr}), \theta \in \mathcal{N}(\theta_{corr})$

```
1: inicializácia B ← 0
2: for all θ_c ∈ N(θ) do
3:    if F(θ_c) > F(θ) then
4:       B ← B + 1
5:    end if
6: end for
7: return B
```

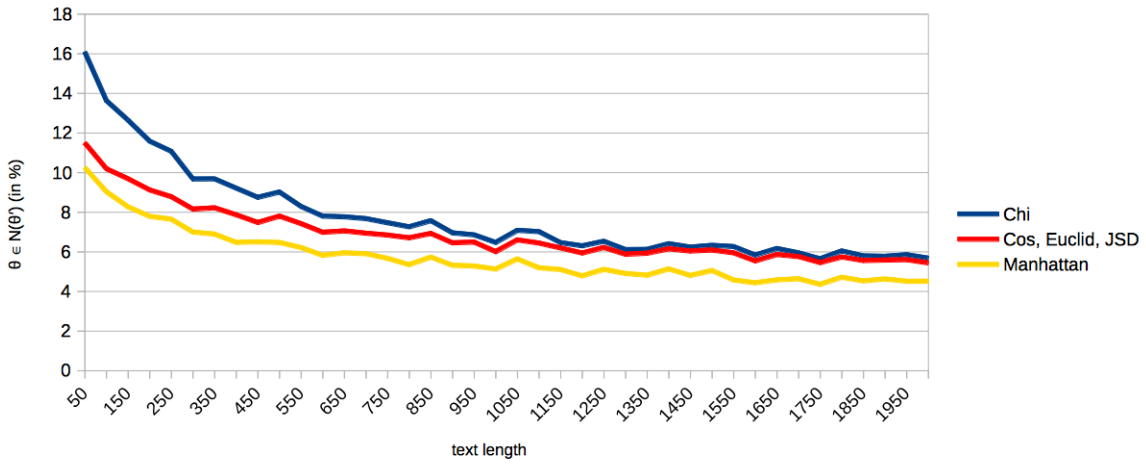
Riešenie θ obsahuje konečný počet susedov, preto sa algoritmus ukončí po vykonaní $\binom{26}{2}$ volaní funkcie $\mathcal{F}$. Predpokladáme, že volanie funkcie $\mathcal{F}$ sa uskutoční v konečnom čase. Výsledok je počet lepšie ohodnotených susedov v intervale $\mathbb{B} \in \langle 0, \binom{26}{2} \rangle$.

Výsledky:
V prípade modelu M_1 (Obr. 4.2) sa ukazuje, že $\mathbb{B}_i > 1, \forall \mathcal{F}_i \in \mathbb{F}$. Najlepšie, avšak rovnako negatívne výsledky dáva funkcia Manhattan.

V prípade modelov M_2 a M_3 (Obr. B.1 a Obr. B.2) počet $\mathbb{B}$ s rastúcou dĺžkou textu prudko klesá. Pre dĺžky nad 300 znakov je už tento počet menej ako 1% (okrem funkcie χ v prípade Markovského modelu 2. rádu). Najmenší počet $\mathbb{B}$ v obidvoch prípadoch dosahuje funkcia JSD.

Pomocou uvedeného experimentu sme zistili, že nájdenie θ_{corr} môže byť problematické, čo ovplyvňuje hlavne voľba modelu jazyka. Voľbou M_i pre $i = 2, 3$ môžeme minimalizovať $\mathbb{B}$. Vyšší počet $\mathbb{B}$ spôsobuje, že výsledok prehľadávania sa nezastaví nájdením θ_{corr}, preto je dôležité ďalej skúmať jeho vzdialenosť od lepších riešení (najbližších extrémov).

Druhá časť experimentu skúma vzdialenosti (počet výpočtových cyklov heuristiky) najbližšieho extrému od θ_{corr}. Na zaručené nájdenie extrému sa používa tzv. adaptive walk, čo môžeme definovať ako postupnosť stavov $(\theta_0, \theta_1, \ldots, \theta_m)$, kde $\forall i < m$, $\mathcal{F}(\theta_i) < \mathcal{F}(\theta_{i+1})$ [40]. Ako adaptive walk sme použili *Steepest Ascent Hill Climbing (ďalej SAHC)* popísaný pomocou Algoritmu 11.



Obr. 4.2: Počet lepšie ohodnotených kľúčov ako správne riešenie pre 1-gramy

Algoritmus 11: Počet výpočtových cyklov SAHC

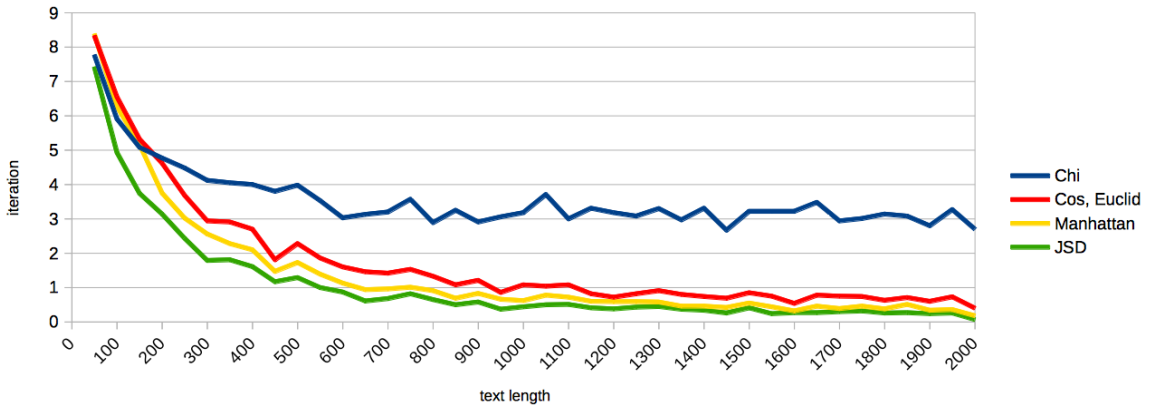
Input: účelová funkcia $\mathcal{F}$
Output: k - počet výpočtových cyklov

1: Inicializácia kroku $k \leftarrow 0$ a počiatočného kandidáta na riešenie θ_k (θ_k = správne riešenie)
2: $f_k \leftarrow \mathcal{F}(\theta_k)$
3: flag $\leftarrow$ **true**
4: **while** (flag = **true**) **do**
5: Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia
6: Výber prvku $\theta_c \in \mathcal{N}(\theta_k)$ s najlepším skóre $f_c \leftarrow \mathcal{F}(\theta_c)$, v prípade viacerých prvkov s rovnakým skóre sa vyberie prvý v poradí
7: **if** $(f_c > f_k)$ **then**
8: Akceptovanie nového riešenia $\theta_{k+1} \leftarrow \theta_c$
9: Ďalší krok $k \leftarrow k + 1$
10: $f_k \leftarrow \mathcal{F}(\theta_k)$
11: **else**
12: flag $\leftarrow$ **false**
13: **end if**
14: **end while**
15: **return** k

Konečnosť algoritmu je zrejmá z toho, že pracujeme na konečnej množine kandidátov. Na tejto množine dosahuje $\mathcal{F}$ maximum. Nakoľko v každom kroku berieme lepšie ohodnotené riešenie, po konečnom počte krokov dosiahneme maximum. Algoritmus vráti počet výpočtových cyklov potrebných k dosiahnutiu lokálneho extrému funkcie $\mathcal{F}$ z počiatočného riešenia.

Výsledky:
Výsledok druhej časti experimentu je vidieť na Obr. B.3, 4.3 a B.4. V prípade M_1 - a to v prípade $\forall \mathcal{F} \in \mathbb{F}$ pre všetky dĺžky textov - je najbližší extrém k θ_{corr} s $\mathcal{F}(\theta) > \mathcal{F}(\theta_{corr})$ ďaleko od θ_{corr}. V prípade M_2 a M_3 táto vzdialenosť s rastúcou dĺžkou textu prudko klesá. V oboch prípadoch dosahuje minimálnu vzdialenosť medzi θ_{corr} a najbližším extrémom funkcia JSD. V prípade M_2 dosahuje od dĺžky textu 300 znakov menej ako 2 swapy a od dĺžky 550 znakov menej ako 1 swap v priemere. V prípade M_3 dosahuje od dĺžky textu 300 znakov menej ako 1 swap.

Z uvedených výsledkov vyplýva, že pri voľbe M_i pre $i = 2, 3$ sa θ_{corr} nachádza blízko lokálneho alebo globálneho extrému. V našom prípade to znamená, že nájdením týchto extrémov môžeme získať takmer správny (zmysluplný) text.



Obr. 4.3: Závislosť vzdialenosti najbližšieho lokálneho extrému s väčším ohodnotením od dĺžky textu (M_2)

Tretia časť experimentu skúma, či sa správne riešenie nenachádza na neutrálnom úseku. Využitá je pritom jednoduchá úprava[9] Algoritmu 10.

Výsledky:
Výsledok tretej časti experimentu je vidieť na Obr. 4.4. V prípade M_1 sme dostali rovnaké hodnoty pre $\forall \mathcal{F} \in \mathbb{F}$ okrem Manhattan (ktorá vykazuje najviac riešení s rovnakým ohodnotením). V prípade M_2 a M_3 sme dostali rovnaké výsledky pre $\forall \mathcal{F} \in \mathbb{F}$. Počet $\theta \in \mathcal{N}(\theta_{corr})$ s rovnakou hodnotou $\mathcal{F}$ s rastúcou dĺžkou textu prudko klesá vo všetkých prípadoch. Od dĺžky 200 znakov je tento počet v prípade M_1 menej ako 5% a v prípade M_2 a M_3 menej ako 2%.

Ak sa pozrieme bližšie na možné príčiny výskytu neutrálnych úsekov (Obr. 4.4), je zrejmé, že neutrálne úseky nie sú priamo závislé od zvolenej účelovej funkcie, ale skôr

[9] Výmenou podmienky $\mathcal{F}(\theta_c) > \mathcal{F}(\theta)$ na $\mathcal{F}(\theta_c) = \mathcal{F}(\theta)$.

od zvoleného modelu jazyka. Keďže všetky funkcie pre konkrétny model dosahujú rovnaké neutrálne úseky okrem jednej výnimky (a to funkcie Manhattan v prípade M_1), rozhodli sme sa sústrediť na také príčiny, ktoré je možné vyvodiť priamo z kvantitatívnej charakteristiky testovacej vzorky.

Nech $\theta = (x_1, x_2, \ldots, x_{26})$ a nech zmena spočíva vo výmene dvoch prvkov x_i, x_j v θ. V prípade Markovského modelu 1. rádu skúmame frekvenciu súradnice $x_i \in \theta$. V prípade modelu 2. rádu skúmame maticu $X = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,26} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,26} \\ \vdots & \vdots & \ddots & \vdots \\ x_{26,1} & x_{26,2} & \cdots & x_{26,26} \end{pmatrix}$, ktorá vyjadruje frekvenciu 2-gramov.

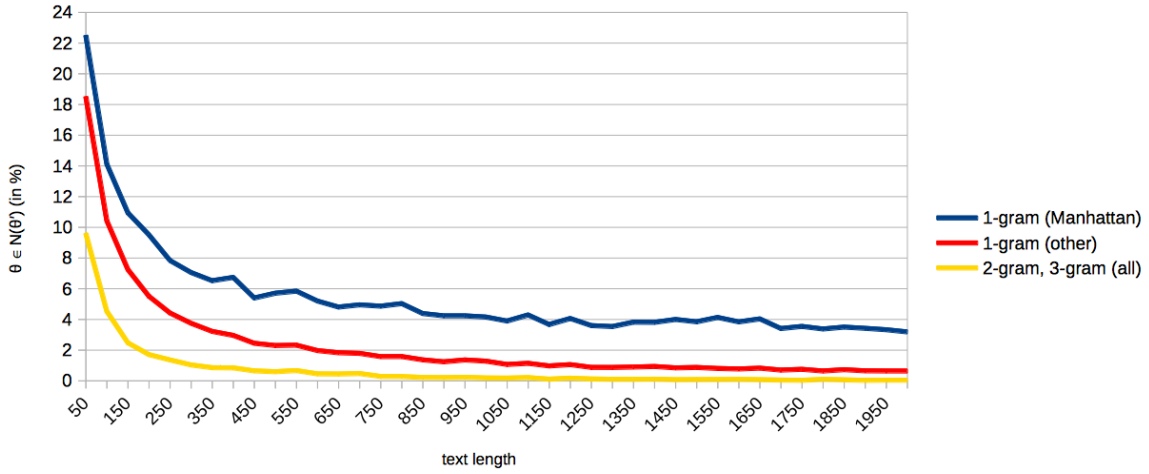
V nasledujúcej časti predpokladáme také funkcie $\mathcal{F}(\theta)$, ktoré vyhodnotia θ po zložkách $\mathcal{F}(\theta) = \sum f(x_i)$ nezávisle.

Zo samotnej štruktúry θ je zrejmé, že v prípade M_1 dostávame rovnakú fitness hodnotu, keď vymeníme prvky $x_i, x_j \in \theta$ s rovnakou frekvenciou. Nech $\{m_1, m_2, \ldots, m_n\}$ označuje počty písmen s rovnakou frekvenciou (napr. m_1 písmen má frekvenciu 1, m_2 písmen má frekvenciu 3, atď.). Potom pre $\forall\theta \in \Theta$ platí, že existuje $\prod_{i=1}^{n} \binom{m_i}{2}$ susedov $\theta_i \in \mathcal{N}(\theta^c)$ s rovnakým ohodnotením ako θ^c. Čiže ľubovoľná výmena prvkov s rovnakou (viď. Obr. 4.5, ktorý zobrazuje závislosť počtu chýbajúcich znakov v kľúči[10] od dĺžky textu) frekvenciou nemení fitness hodnotu θ^c.

Výmena dvoch prvkov $x_i, x_j \in \theta$ v prípade M_2 spôsobuje zmenu dvoch riadkov ako aj stĺpcov i, j v matici X. Rovnakú fitness hodnotu je možné dostať napr. v takom prípade, keď výmenou x_i, x_j sa nemení matica X. Táto situácia nastáva (najpravdepodobnejšie) v prípade, keď frekvencia x_i a x_j je nulová (všetky prvky $x_{i,*}$ a $x_{*,i}$ sú v X nulové.). V prípade m znakov s nulovou frekvenciou pre $\forall\theta \in \Theta$ platí, že existuje $\binom{m}{2}$ susedov $\theta_i \in \mathcal{N}(\theta^c)$ s rovnakým ohodnotením ako θ^c. Keďže frekvencia jedného znaku ovplyvňuje (a je ovplyvnený) výskyt viacerých dvojíc[11], výmena dvoch prvkov s rovnakou ale nenulovou frekvenciou vedie v tomto prípade k zmene matice X.

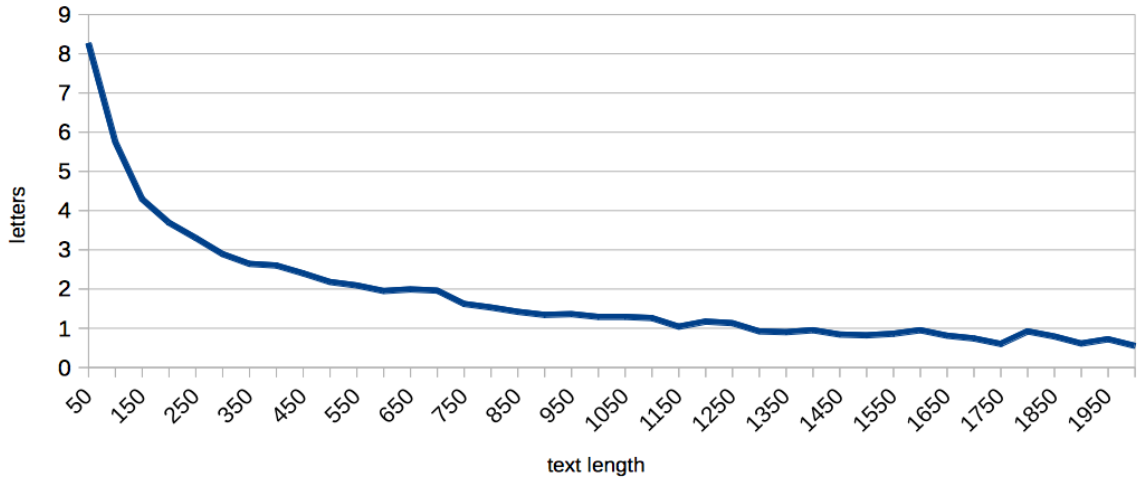
[10] Ako aj v *OT*.

[11] Presnejšie 51 referenčných hodnôt, napr. písmeno 'a' ovplyvňuje frekvenciu dvojíc $\{aa, ab, \ldots az, aa, ba, \ldots za\}$. Dvojicu 'aa' treba zarátať len raz.



Obr. 4.4: Počty rovnako ohodnotených susedných riešení ako správne riešenie pre n-gramy s $n = 1, 2, 3$

Pomocou tejto časti experimentu sme poukázali na existenciu neutrálnych úsekov, ktoré sťažujú nájdenie správneho riešenia. Najväčšie neutrálne úseky boli nájdené v prípade M_1, v prípade ktorého sme ozrejmili, že ich príčinou sú hlavne písmená s rovnakou frekvenciou. Neutrálne úseky zapríčinené písmenami s nulovou frekvenciou, ktoré môžu vzniknúť v prípade oboch modelov však nepredstavujú problém. Riešenia na takomto neutrálnom úseku predstavujú rovnako dobré riešenie.



Obr. 4.5: Priemerný počet chýbajúcich písmen v 26 prvkovom kľúči

Z celkových výsledkov vyplýva, že v prípade M_1 žiadna $\mathcal{F} \in \mathbb{F}$ nevyhovuje našim požiadavkám, t.j. $\mathcal{F}(\theta_{corr}) \neq \mathcal{F}_{max}$, okrem toho θ_{corr} nie je dostatočne blízko k lokál-

nym/globálnym extrémom a samotný proces prehľadávania je sťažený veľkým počtom neutrálnych úsekov. Tieto nedostatky však môžu byť minimalizované voľbou modelov jazyka vyššieho rádu. Z výsledkov taktiež vyplýva, že najvhodnejšia metóda pre vyššie modely jazyka je funkcia JSD. Treba však poznamenať, že zvolené funkcie vo všeobecnosti nespĺňajú požiadavku aby správne riešenie malo najlepšie ohodnotenie, čo predstavuje problém hlavne v prípade krátkych textov.

Ohodnotenie kandidátov podľa percentuálnej zhody grafém výsledku a správneho riešenia

Zamerali sme sa na overenie kvality najlepších výsledkov dosiahnutých lúštením prostredníctvom meta-heuristík. Kvalitu výsledkov lúštenia sme interpretovali na základe percentuálnej zhody grafém θ s θ_{corr}, ktoré sme už skôr označili ako $\mathbb{M}(\theta)$. Parameter θ budeme v označovaní vynechávať, ak budú jasné súvislosti. Dosiahnutie správneho riešenia závisí okrem účelovej funkcie aj od voľby modelu jazyka a od samotnej meta-heuristiky, preto je potrebné skúmať lúštenie v závislosti od uvedených troch premenných.

Experiment spočíva v lúštení a vyhodnotení ZT jednotlivých dĺžok (podľa sekcie 4.2.1) pre $\mathcal{F} \in \mathbb{F}$. Experiment je popísaný pomocou Procedúry 12.

Procedúra 12: Postup vykonania experimentu

Input: účelová funkcia $\mathcal{F}^a$
Input: ZT a prislúchajúci OT
1: základné heuristiky[b] $H \leftarrow \{\text{HC, TS}^c\text{, SA}^d\}$
2: rády Markovského modelu $M \leftarrow \{1, 2\}$
3: **for all** $m \in M$ **do**
4: **for all** $h \in H$ **do**
5: lúštenie: $\theta_c \leftarrow h(\mathcal{F}, m, ZT)$
6: $X_c \leftarrow d_k(ZT, \theta_c)$
7: vypočítaj zhodu X_c s X
8: **end for**
9: **end for**

[a] Jednotlivé $\mathcal{F}$ sme pretransformovali na maximalizačný problém. Okrem funkcie Cos je cieľom minimalizovať hodnotu funkcie na 0 z hodnoty $x \in \mathbb{R}^+$. Vo vybraných meta-heuristikách sme teda použili hodnotu $-\mathcal{F}$. V prípade Cos sme použili hodnotu $-(1-\mathcal{F})$.

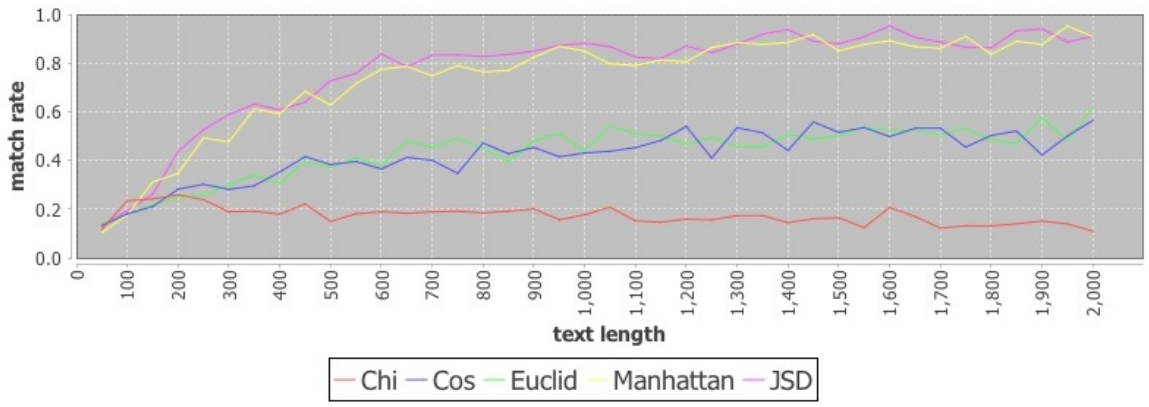
[b] Empiricky sme zistili, že metódy konvergujú rýchlo, maximálny počet výpočtových cyklov (ukončovaciu podmienku) sme nastavili na $MI = 5000$.

[c] S veľkosťou *tabuList* $= 100$.

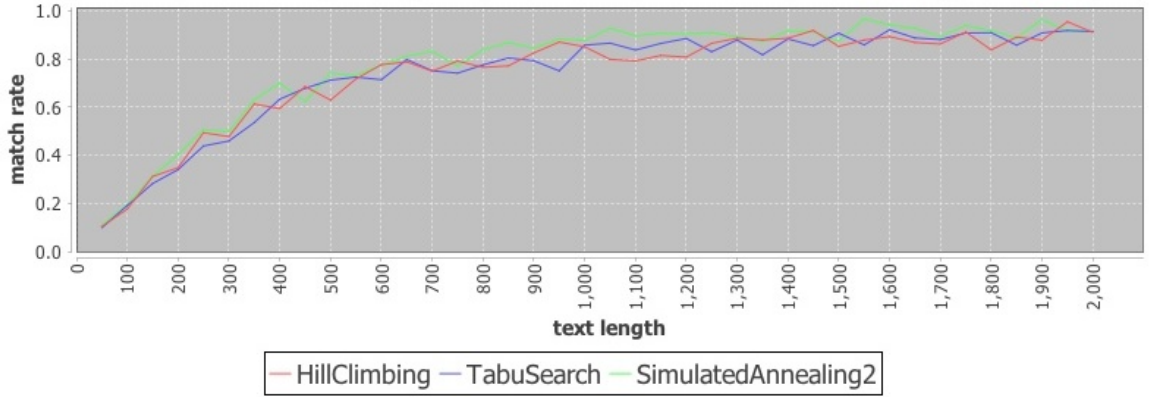
[d] $\delta = 0.5$ a $k_\delta = \frac{MI}{2}$.

Výsledky:
V prípade M_1 (Obr. B.5, B.6 a B.7) je úspešnosť $\forall \mathcal{F} \in \mathbb{F}$ porovnateľná nezávisle od

voľby heuristiky. Nakoľko sa v priemere nedosahuje ani $\mathbb{M} = 50\%$, môžeme skonštatovať, že M_1 nie je postačujúci. Treba však dodať, že v niektorých špeciálnych prípadoch, keď neberieme do úvahy priemerné ale najlepšie výsledky, je možné dosiahnuť až $\mathbb{M} = 90\%$ (Obr. B.8, B.9 a B.10).
V prípade M_2 (Obr. 4.6, B.11 a B.12) je možné pre dlhšie texty (cca 600 a viac znakov) dosiahnuť v priemere až $\mathbb{M} = 80\%$. Z toho, že výsledky testovania jednotlivých heuristík sú veľmi podobné (napr. Obr. 4.7 a B.16) vyplýva, že **úspešnosť' lúštenia nezávisí od voľby heuristiky**. Grafy porovnávajúce priemernú úspešnosť' lúštenia pri rôznych $\mathcal{F}$ a zafixovanej meta-heuristiky (Obr. 4.6, B.11 a B.12) ukazujú, že **úspešnosť' lúštenia je závislá na voľbe $\mathcal{F}$**. Z výsledkov vyplýva, že najväčšiu priemernú $\mathbb{M}$ dosahujú funkcie JSD a Manhattan (okolo 90%). Funkcie Cos a Euklid dosahujú v priemere okolo $\mathbb{M} = 50\%$. Funkcia χ^2 dosahuje najhoršie výsledky. Z Obr. B.13, B.14 a B.15 je vidieť, že už pri textoch dĺžky 200 znakov je možné dosiahnuť' $\mathbb{M} = 100\%$ v prípade každej $\mathcal{F}$, okrem funkcie χ^2.



Obr. 4.6: Priemerná hodnota $\mathbb{M}$ pre 2-gramy v prípade HC

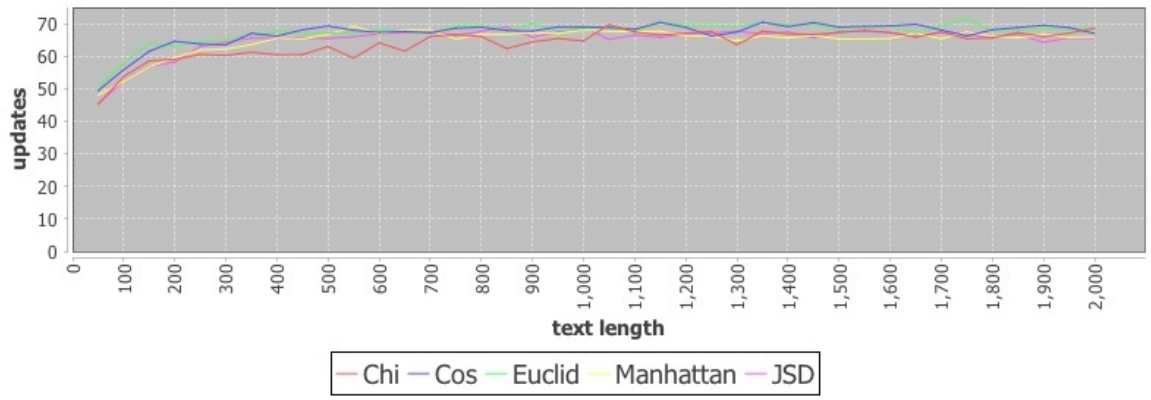


Obr. 4.7: Priemerná hodnota $\mathbb{M}$ pre HC, TS a SA pre 2-gramy v prípade funkcie Manhattan

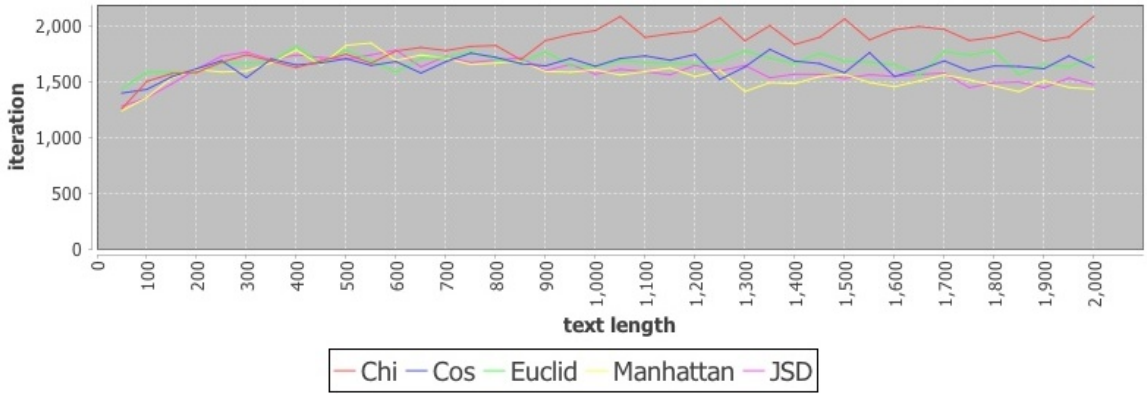
Poznámka 4 *Okrem vyššie uvedených sme skúmali aj možnosť kombinácie rôznych M_i pri konštrukcii $\mathcal{F}$ (podobne ako A. Clark v [13]). Experimenty boli vykonané kvôli porovnateľnosti s výsledkami z [13] pre funkciu Manhattan. Zamerali sme sa na použitie M_1 až M_3 (1-gramy $\mathcal{F}_{M_1}$, 2-gramy $\mathcal{F}_{M_2}$ a 3-gramy $\mathcal{F}_{M_3}$) s dodatočným slovníkovým ohodnotením $\mathcal{F}_{slovnik}$. Ako slovníkové ohodnotenie sme použili percentuálne pokrytie textu slovami zo slovníka. Pri konštrukcii boli vyskúšané rôzne váhy pre jednotlivé časti účelovej funkcie $\mathcal{F}(\theta) = \alpha \cdot \mathcal{F}_{M_1}(\theta) + \beta \cdot \mathcal{F}_{M_2}(\theta) + \gamma \cdot \mathcal{F}_{M_3}(\theta) + \delta \cdot \mathcal{F}_{slovnik}(\theta)$. Kombinácia rôznych modelov s pridaním slovníkového ohodnotenia však nepriniesla žiadne vylepšenie, preto ich v práci neuvádzame.*

Z výsledkov vyššie popísaných experimentov je zrejmé, že $\mathcal{F}$ a M_i, ktoré umožnia dosiahnutie najlepších výsledkov vieme presne určiť. Ako bolo spomenuté, voľba heuristiky z tohto hľadiska nie je podstatná. Dôležitý parameter ale môže predstavovať výpočtová náročnosť vybranej meta-heuristiky, ktorú môžeme bližšie špecifikovať počtom volaní $\mathcal{F}$. Z toho dôvodu sme sa rozhodli experimentálne zistiť:
a) priemerný počet volaní $\mathcal{F}$ potrebných na dosiahnutie konečného výsledku;
b) počet zmien, ktoré prispievajú k vylepšeniu kandidáta
v prípade využitia rôznych meta-heuristík. Keďže v prípade M_1 nebola preukázaná dostatočná úspešnosť, v experimentoch sme sa zamerali len na M_2.

Výsledky:
Ako ukazuje Obr. B.18 a B.17, v prípade HC nepotrebujeme ani 2500 iterácií k vykonaniu poslednej zmeny, počas ktorej sa na vylepšenie kandidáta vykoná v priemere 70 zmien. V prípade TS (Obr. 4.9 a 4.8) na dosiahnutie rovnakých výsledkov stačí v priemere menej ako 2000 iterácií. Napriek tomu, že všetky účelové funkcie sú porovnateľné, funkcie JSD a Manhattan konvergujú k finálnemu riešeniu najrýchlejšie a to v prípade každej heuristiky.



Obr. 4.8: Priemerný počet vylepšení pre 2-gramy v prípade TS



Obr. 4.9: Priemerná hodnota iterácie poslednej zmeny pre 2-gramy v prípade TS

Ako zhodnotenie vykonaných experimentov môžeme konštatovať, že správnou voľbou účelovej funkcie, modelu jazyka a meta-heuristiky je možné skonštruovať efektívnu metodiku na lúštenie monoalfabetickej substitúcie. Napriek všeobecnej úspešnosti problémom zostáva lúštenie krátkych správ do 500 znakov (viď. obrázky sekcie 4.2.2). Bližšou analýzou tejto otázky sa zaoberáme v nasledujúcej sekcii.

4.3 Lúštenie krátkych šifrovaných správ

Na presnejšie určenie príčin zlyhania lúštenia krátkych šifrovaných správ sme sa rozhodli zamerať sa na analýzu ďalších vlastností účelových funkcií. Analýzou globálnej geometrie prehľadávacieho priestoru - tzv. fitness landscape sa pokúsime o nájdenie dodatočných opatrení, ktoré majú potenciál vylepšiť úspešnosť lúštenia krátkych správ. Pomocou tejto analýzy môžeme správne vybrať a nastaviť vhodné meta-heuristiky ako aj ich parametre [40].

V tejto kapitole budeme experimentálne skúmať také vlastnosti účelových funkcií, ktoré sú považované [79, 40] za najdôležitejšie, t.j. na počet a umiestnenie lokálnych a globálnych extrémov a na homogenitu extrémov. Na základe našich predchádzajúcich výsledkov sa sústredíme len na účelové funkcie s najväčšou úspešnosťou - na JSD a Manhattan.

V experimentoch sme sa najprv zameriavali na zistenie počtu lokálnych extrémov, ktorý vyjadruje obtiažnosť dosiahnutia globálneho extrému. Priame získanie, alebo odhad počtu lokálnych a globálnych extrémov nie je vo všeobecnosti jednoduché. Na odhad počtu extrémov sa používajú metódy založené na opakovanom spúštaní[12] (reinicializácia) zvolenej meta-heuristiky. V prípade, že počet reinicializácie r podstatne presahuje počet dosiahnutých extrémov k môžeme predpokladať, že sme našli väčšinu lokálnych extrémov, ako aj globálny extrém [79, 40]. Ak hodnota k nie je výrazne nižšie ako r,

[12]Pri opakovanom spúštaní sa počiatočné riešenia inicializujú náhodne.

musíme predpokladať existenciu viacero ďalších extrémov [79, 40].
Na získanie dostatočného množstva údajov sme sa rozhodli lúštenie zopakovať $r = 1000$ krát pre každú testovaciu vzorku. V experimentoch sme použili *SAHC* s reinicializáciou [79] (popísaný v Algoritme 13).

Algoritmus 13: SAHC s reinicializáciou

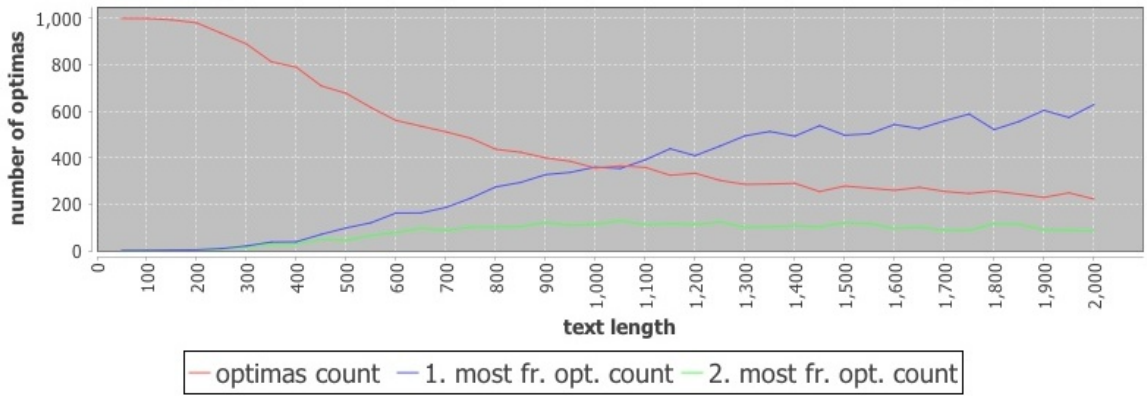
Input: r_{max} - počet reinicializácií
Output: R - dosiahnuté extrémy

$R \leftarrow \emptyset$
for $r \leftarrow 1$ **to** r_{max} **do**
  Inicializácia počiatočného kandidáta na riešenie θ_k a inicializácia kroku $k \leftarrow 0$;
  $f_k \leftarrow \mathcal{F}(\theta_k)$
  flag $\leftarrow$ **true**
  while (flag = **true**) **do**
    Nájdenie susedov $\mathcal{N}(\theta_k)$ aktuálneho riešenia
    Výber prvku $\theta^c \in \mathcal{N}(\theta_k)$ s najlepším skóre $f_c \leftarrow \mathcal{F}(\theta_c)$
    if $(f_c > f_k)$ **then**
      Akceptovanie nového riešenia $\theta_{k+1} \leftarrow \theta^c$
      Ďalší krok $k \leftarrow k + 1$.
    else
      $R \leftarrow R \cup \{\theta_{k+1}\}$
      flag $\leftarrow$ **false**
    end if
  end while
end for

Principiálne sa jedná o opätovné volanie Algoritmu 11 (r_{max}-krát).

Výsledky:
Ako výsledok uvádzame priemerný počet dosiahnutých extrémov k, ako aj priemernú početnosť prvých dvoch najfrekventovanejších extrémov v závislosti od zvolenej dĺžky testovacej vzorky. V prípade M_1 (Obr. B.20 a B.19) funkcia Manhattan vykazuje veľký počet rôznych extrémov a to v prípade krátkych aj dlhých textov. Z toho môžeme usúdiť, že nájdenie globálneho extrému je v tomto prípade obtiažne. Funkcia JSD v porovnaní s funkciou Manhattan vykazuje lepšiu charakteristiku, pravdepodobnosť dosiahnutia globálneho extrému s rastúcou dĺžkou textu narastá.

Z Obr. 4.10 a B.21 je vidieť, že pre M_2 vykazujú obe funkcie podobné výsledky. V prípade textov do dĺžky 200 znakov sa hodnota k rovná hodnote r. Od dĺžky okolo 200 znakov (čo je hraničná hodnota reprezentatívnej dĺžky textu pre 2-gramy) k vykazuje klesajúcu tendenciu, čiže rastie pravdepodobnosť nájdenia globálneho extrému. Týmto faktom môžeme zdôvodniť aj skutočnosť, že v prípade predošlých experimentov (sekcia 4.2.2) bola vysoká priemerná úspešnosť v prípade dlhších textov. Na základe našich výsledkov môžeme skonštatovať, že v prípade krátkych textov je nutné dodatočne riešiť problém veľkého množstva lokálnych extrémov.



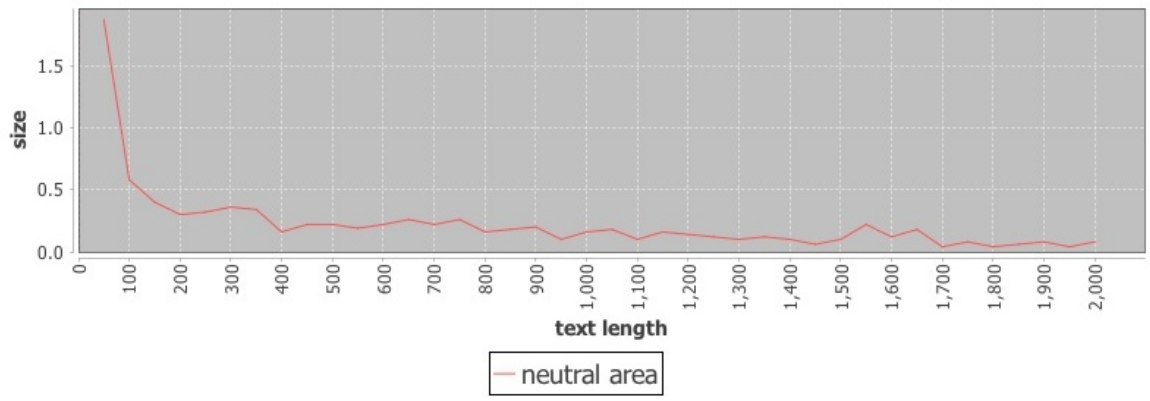
Obr. 4.10: Priemerný počet dosiahnutých optím a kardinalita najpočetnejších optím (JSD, 2-gramy)

V predchádzajúcom experimente sme ukázali, že v prípade krátkych textov máme viac lokálnych/globálnych extrémov - jedná sa o tzv. multimodálny problém. Pri tomto veľkom počte extrémov situáciu môže skomplikovať aj homogenita výsledkov, t.j. výskyt viacerých rôznych extrémov s rovnakou fitness hodnotou[13]. V ďalšom experimente sme porovnali fitness hodnotu extrémov získaných v predošlom experimente a skúmali sme homogénnu časť s najväčším počtom rovnako ohodnotených extrémov (ďalej len $hMax$).

Výsledky:
V prípade M_1 (Obr. B.23 a B.22) funkcia JSD vykazuje prudký pokles veľkosti $hMax$, ktorý od dĺžky 200 znakov v sebe zahŕňa menej ako 20% extrémov a od dĺžky 1000 znakov je minimálny. V prípade funkcie Manhattan však táto veľkosť ostáva okolo 40% v prípade textov dĺžky 2000 znakov. Na základe týchto zistení môžeme povedať, že v prípade krátkych textov je vysoká pravdepodobnosť, že nájdený extrém je súčasťou homogénneho alebo neutrálneho úseku.

V prípade M_2 (Obr. 4.11 a B.24) sú získané hodnoty porovnateľné pre obe skúmané funkcie. Už v prípade krátkych textov (100 - 200 znakov) je veľkosť $hMax$ minimálna. To znamená, že v prípade M_2 rôzne extrémy majú odlišné hodnoty, čiže prehľadávanie nie je sťažené homogénnymi alebo neutrálnymi úsekmi.

[13]Principiálne sa jedná o analýzu neutrality lokálnych extrémov, ktoré však nemusia byť susediace riešenia.



Obr. 4.11: Priemerná veľkosť najväčšieho neutrálneho úseku (JSD, 2-gramy)

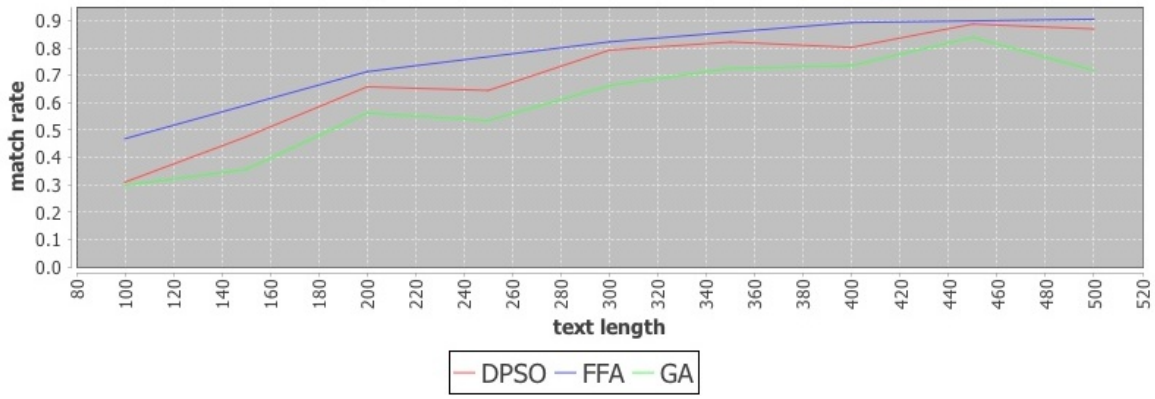
Z doterajších výsledkov vyplýva, že v prípade M_1 existuje veľký počet rôznych lokálnych extrémov, z ktorých má veľká časť rovnaké ohodnotenie. Kvôli tomu je problematické navrhnúť vhodné vylepšenie na zvýšenie úspešnosti lúštenia. V prípade M_2 počet rôznych lokálnych extrémov prudko klesá s rastom dĺžky testovacej vzorky a prehľadávanie s veľkou pravdepodobnosťou nie je obmedzené výskytom neutrálnych úsekov. Veľký počet lokálnych extrémov v tomto prípade ale môže spôsobovať zaseknutie (t.j. nie sme schopný opustiť aktuálny extrém a nájsť tak lepšie riešenie). Na základe uvedených zistení môžeme povedať, že na riešenie problémov pri lúštení krátkych textov je potrebné si zvoliť špeciálnu techniku, ktorá umožní opustenie lokálnych extrémov v prípade zaseknutia.

Prvá možnosť je použitie špeciálnych *prírodou inšpirovaných meta-heuristík*, ktoré sú navrhnuté na riešenie optimalizačných problémov s veľkým počtom lokálnych extrémov. Experimentálne sme overili úspešnosť vybraných prírodou inšpirovaných metód. Zamerali sme sa na GA, PSO a FFA. V prípade PSO a FFA sme použili diskrétne verzie, uvedené v sekcii 3.4.3. Tieto metódy však obsahujú aj také parametre, od nastavenia ktorých značne závisí úspešnosť prehľadávania. Tieto parametre sú veľkosť populácie, počet iterácií a iné parametre špecifické pre danú meta-heuristiku.
V prípade M_1 (Obr. B.25 a B.26) sme skúmali rôzne nastavenia, avšak žiadnym z nich sme nedosiahli v priemere viac ako $\mathbb{M} = 30\%$;
V prípade M_2 (Obr. 4.12 a B.27) uvádzame výsledky našich experimentov pre najlepšie nastavenia parametrov, ku ktorým sme sa dopracovali empirickým skúšaním:

- FFA: 200 iterácií, veľkosť populácie 150, $\gamma = 0.1$, $\alpha = 0.2$
- DPSO: 200 iterácií, veľkosť populácie 500, $c_1 = 0.6$, $c_2 = 0.8$
- GA: 2000 iterácií, veľkosť populácie[14] 20 (elitarizmus bez zmeny - 1 jedinec; elitarizmus s mutáciou - 4 jedinci; náhodný výber s mutáciou - 5 jedincov; turnajový výber s mutáciou - 5 jedincov; ruletový výber s mutácia - 5 jedincov)

[14]Zistili sme, že kríženie neprináša žiadne vylepšenie.

Z výsledkov môžeme konštatovať, že prírodou inšpirované meta-heuristiky (v prípade M_2) vykazujú značné vylepšenie úspešnosti v porovnaní s jednoduchými meta-heuristikami a to v prípade oboch funkcií. Najlepšie výsledky sme dosiahli s funkciou JSD. V prípade FFA môžeme dosiahnuť až $\mathbb{M} = 80\%$ už od dĺžky 250 znakov a v prípade DPSO od dĺžky 300 znakov. O niečo horšie výsledky, ako v prípade JSD sme dosiahli pomocou funkcie Manhattan.



Obr. 4.12: Priemerná hodnota $\mathbb{M}$ pre DPSO, FFA a GA (JSD, 2-gramy)

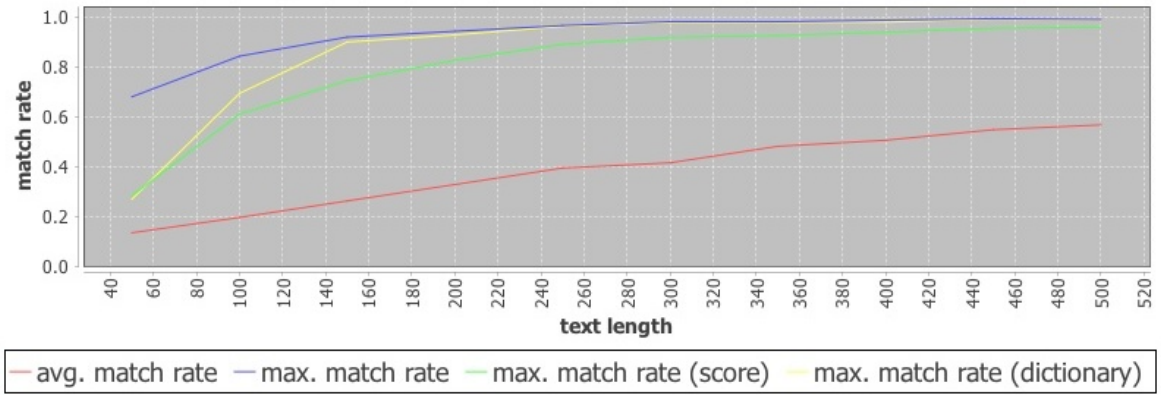
Ďalším spôsobom, ako riešiť problém zaseknutia v lokálnych extrémoch je *reinicializácia* - znovu spustenie prehľadávacieho procesu. Na zaručenie získanie optima sme použili SAHC. Lúštenie sme zopakovali $r = 1000$ krát pre každú testovaciu vzorku, kde sme všetky dosiahnuté výsledky (extrémy) uložili. Následne sme analyzovali množinu dosiahnutých extrémov.
V Obr. 4.13 uvádzame priemerné hodnoty výsledkov experimentov (v prípade každej dĺžky textu uvádzame priemer pre $t = 100$ rôznych vzoriek):

- avg. match rate - priemerná $\mathbb{M}(\theta)$. Uvádzame priemer 1000 (r) krát 100 (t) výsledkov pre každú dĺžku.

- max. match rate - maximálna $\mathbb{M}(\theta)$. Uvádzame priemer výsledkov pre každú dĺžku zo 100 (t) rôznych textov, kde pre každý text bola vybratá maximálna hodnota z 1000 (r) reinicializácií.

- max. match rate (score) - maximálna $\mathbb{M}(\theta)$, pre θ zvolených podľa najlepšej fitness hodnoty. Uvádzame priemer výsledkov pre každú dĺžku zo 100 (t) rôznych textov, kde pre každý text bolo vybraté jedno θ s najlepšou fitness hodnotou z 1000 (r) reinicializácií.

- max. match rate (dictionary) - maximálna $\mathbb{M}(\theta)$, pre θ zvolených podľa slovníkového ohodnotenia. Uvádzame priemer výsledkov pre každú dĺžku zo 100 (t) rôznych textov, kde pre každý text bolo vybraté jedno θ s najlepšou fitness hodnotou z 1000 (r) reinicializácií.



Obr. 4.13: Priemerná hodnota $\mathbb{M}$ pri reinicializácií pre JSD

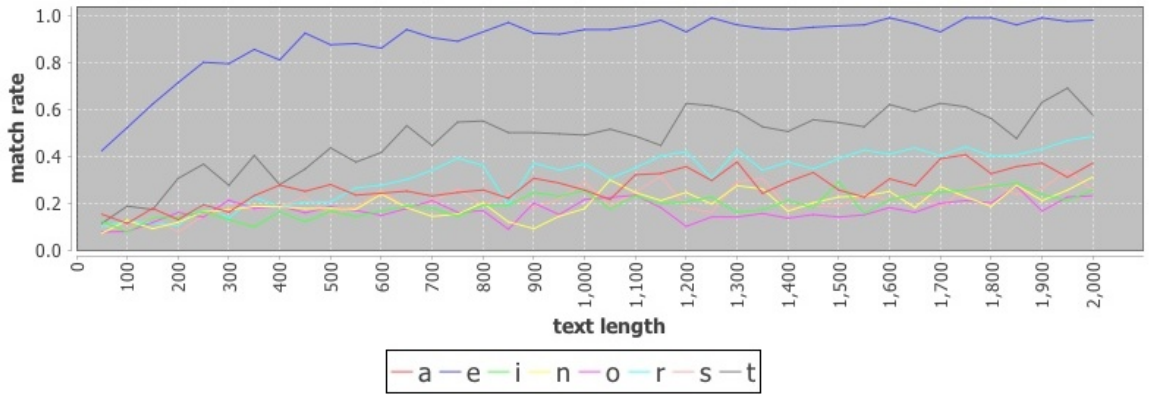
Z Obr. 4.13 je vidieť, že reinicializácia a následný výber najlepších riešení značne vylepšuje úspešnosť. Pomocou reinicializácie je možné dosiahnuť $\mathbb{M} = 80\%$ už pre minimálnu stanovenú dĺžku 200 znakov. Ďalej môžeme maximalizovať pravdepodobnosť nájdenia správneho výsledku v prípade dlhších správ nad 300 znakov na približne 100%. V Obr. B.28, B.30, B.29 a B.31 uvádzame výsledky lúštenia s reinicializáciou pre M_1 a M_2 aj pre dlhšie testovacie vzorky. Z výsledkov vyplýva, že v prípade M_2 reinicializácia predstavuje značné vylepšenie úspešnosti lúštenia pre funkcie Manthattan a JSD. V prípade M_1 však neprináša žiadne podstatné vylepšenie.

Zaujímavou otázkou je aj samotná štruktúra dosiahnutých lokálnych a globálnych extrémov z pohľadu korektnosti písmen v kľúči. Vychádzali sme z výsledkov predošlého experimentu. V prípade každej dĺžky textu sme získali 10^5 výsledkov, kde sme skúmali jednotlivé pravdepodobnosti, ktorými sú písmená správne nájdené v kľúči.

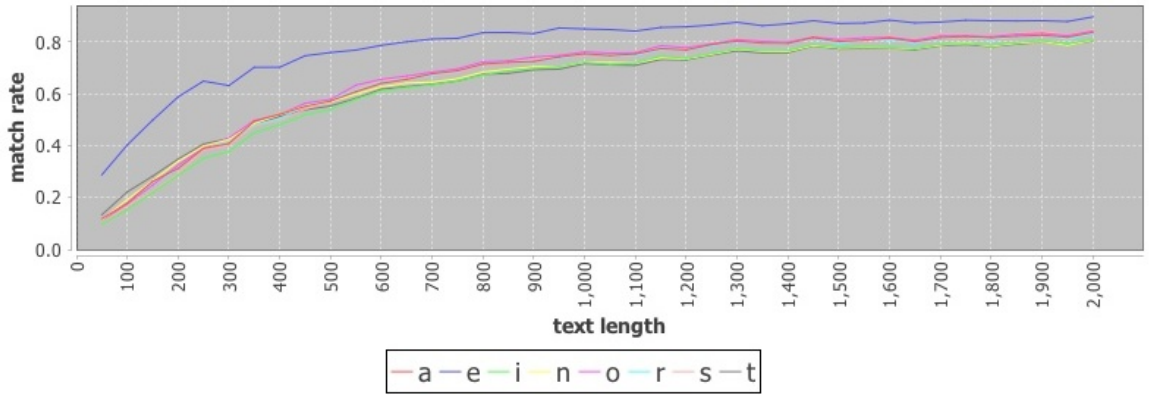
Písmená abecedy sme podľa ich frekvencie výskytu v jazyku rozdelili na tri skupiny: viac ako 5% výskytu, medzi 1 a 5% výskytu a menej ako 1% výskytu. V experimente sme použili najlepšiu funkciu JSD.

Výsledky:
Zistili sme, že najčastejšie sa správne určia najfrekventovanejšie písmená abecedy (Obr. 4.14 a 4.15). Písmeno **e** sme určili správne s najväčšou pravdepodobnosťou - v prípade M_1 je to skoro 100%, kde nasleduje písmeno **t** s okolo 50%-ami. Pravdepodobnosť správneho nájdenia ostatných písmen je menšia.V prípade M_2 je možné správne určiť písmeno **e** s 80-90%-nou pravdepodobnosťou. Pravdepodobnosť správneho nájdenia ostatných frekventovaných písmen je viac-menej rovnaká.

Stredne a málo frekventované písmená sa určia správne v prípade M_1 (Obr. B.32 a B.33) do 50% pravdepodobnosťou. Pri M_2 (Obr. B.34 a B.35) s 80% pravdepodobnosťou, ale len v prípade dlhších textov.



Obr. 4.14: Pravdepodobnosť nájdenia najviac frekventovaných písmen (1-gramy a JSD)



Obr. 4.15: Pravdepodobnosť nájdenia najviac frekventovaných písmen (2-gramy a JSD)

Z vykonaných experimentov sme zistili, že pri lúštení krátkych textov je dôležitým zdrojom informácií analýza fitness landscape. Zistili sme, že v prípade M_1 veľký počet lokálnych extrémov a vysoká miera výskytu homogénnych úsekov neumožňujú vylepšenie úspešnosti lúštenia. Jediné písmeno, ktoré je možné správne nájsť s vysokou pravdepodobnosťou je písmeno e. V prípade M_2 je možné zvýšiť úspešnosť lúštenia pomocou využitia prírodou inšpirovaných meta-heuristík. Tieto vylepšenia však zaostávajú v porovnaní s výsledkami dosiahnutými pomocou reinicializácie. Úspešnosť lúštenia sme dokázali zvýšiť na viac ako $\mathbb{M} = 90\%$ aj pri 200 znakových krátkych textov. Táto metóda je taktiež vhodná na maximalizáciu úspešnosti v prípade dlhších textoch. Pre komplexný pohľad na problematiku lúštenia krátkych textov sme sekciu doplnili o analýzu pravdepodobnosti správneho určenia písmen.

Kapitola 5

Lúštenie homofónnej substitúcie pomocou meta-heuristík

V tejto kapitole sa zaoberáme možnosťami lúštenia homofónnej substitúcie pomocou meta-heuristík. Špecifickou charakteristikou tejto šifry je, že je skonštruovaná tak, aby odstraňovala nedostatky monoalfabetickej substitúcie. Najefektívnejšou metódou lúštenia monoalfabetickej substitúcie je využitie meta-heuristík, ktoré sa však doposiaľ nepodarilo dostatočne efektívne aplikovať pri lúštení homofónnej substitúcie. Z uvedených dôvodov sme sa rozhodli hlbšie preskúmať túto problematiku.

V prvej časti kapitoly uvádzame poznatky o vlastnostiach homofónnej substitúcie vyplývajúcich z konštrukcie šifry, ktoré sa následne pokúsime využiť pri jej lúštení. Pri lúštení vychádzame z transformácie kryptoanalýzy na optimalizačný problém. V sekcii 5.3 skúmame možnosti reprezentácie homofónnej substitúcie ako optimalizačného problému. Na záver v sekcii 5.3.5 uvádzame lúštenie špeciálneho prípadu homofónnej substitúcie na základe cyklického opakovania znakov v texte.

5.1 Špecifikácia homofónnej substitúcie

Pod pojmom *homofónna šifra* sa rozumie znáhodnená šifra, kde frekvencia znakov zašifrovaného textu sa vyrovnáva za účelom získania rovnomerného rozdelenia znakov [29]. Tento postup slúži na znemožnenie útokov, založených na štatistických vlastnostiach textu. Homofónna šifra teda popisuje šifrovacie algoritmy, kde výskyt znakov ZT je rovnomerne rozdelený.
V tejto práci pod *homofónnou substitúciou* chápeme substitučnú šifru, ktorá je homofónna šifra a platí, že znaky z $\mathcal{A}_P$ sa šifrujú na niekoľko rôznych znakov z $\mathcal{A}_C$. Pojem *homofón* sa používa na označenie podmnožiny znakov z $\mathcal{A}_C$, na ktoré sa šifruje jeden konkrétny znak z $\mathcal{A}_P$.

Homofónnu substitúciu môžeme popísať pomocou zadefinovaného kryptosystému (Definícia 1), avšak kvôli presnejšiemu vyjadreniu štruktúry homofónnej substitúcie mô-

žeme túto definíciu rozšíriť o trojicu $\mathcal{H}$, $\mathcal{L}$ a $\mathcal{M}$, kde: [45]

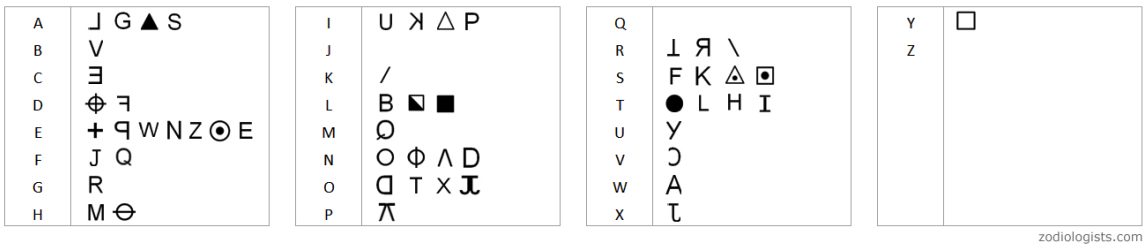
- $H_i \in \mathcal{H}$ sú množiny homofónov (t.j. $H_i \subset \mathcal{A}_C$). Množiny H_i sú disjunktné. Pre každé $y_j \in H_i$ platí, že $d_k(h_j) = x_i$ a $x_i \in \mathcal{A}_P$.

- $\mathcal{L}$ je množina veľkostí množín homofónov. Pre $l_i \in \mathcal{L}$ platí, že $l_i = |H_i|$ je počet možných spôsobov šifrovania znaku $p_i \in \mathcal{A}_P$ na znaky z $\mathcal{A}_C$ a zároveň platí, že $l_1 + l_2 + \cdots + l_t = |\mathcal{A}_C|$, kde $t = |\mathcal{A}_P|$.

- $\mathcal{M}$ vyjadruje množstvo použitých homofónov vzhľadom na dĺžku zašifrovaného textu, $\mathcal{M} = \frac{L}{N}$, kde N je dĺžka zašifrovaného textu a $L \leq |\mathcal{A}_C|$ je počet použitých znakov z $\mathcal{A}_C$. Platí pritom, že $\frac{1}{N} \leq \mathcal{M} \leq 1.0$.

Homofónnu substitúciu môžeme charakterizovať pomocou parametra $\mathcal{M}$. Z hodnoty parametra $\mathcal{M}$ priamo vyplýva (za predpokladu, že šifra bola skonštruovaná podľa Algoritmu 14) zložitosť homofónnej substitúcie. Čím vyššia je hodnota $\mathcal{M}$, tým je lúštenie komplikovanejšie [45].

Dobrým príkladom homofónnej substitúcie je Zodiac-ova rozlúštená šifra [116]. Na obrázku 5.1 je ukážka prvej (rozlúštenej) šifry označovanej ako Z408, ktorá pozostáva z textu dlhého 408 znakov a obsahuje 54 neznámych symbolov. Zložitosť Z408 je v tomto prípade $\mathcal{M} = 0.132$.



Obr. 5.1: Kľúč homofónnej substitúcie Z408 (prevzaté z [116])

Ak chceme vytvoriť homofónnu substitúciu s parametrami N a $\mathcal{M}$, môžeme použiť Algoritmus 14 (prevzatý z [45]), ktorý pre dané parametre generuje (čo najviac) rovnomerné rozdelenie znakov. Jedná sa teda o najzložitejší prípad homofónnej substitúcie vzhľadom na $\mathcal{M}$ [45].

Algoritmus 14: Generovanie veľkosti množín homofónov

1: Pre všetky $x_i \in \mathcal{P}$
2: $l_{x_i} \leftarrow \max\{1, \lfloor p(x_i) \times N \times \mathcal{M} \rfloor\}$

$p(x_i)$ označuje frekvenciu výskytu znakov OT pre $x_i \in \mathcal{P}$; N označuje dĺžku textu.

5.2 Konštrukcia homofónnej substitúcie

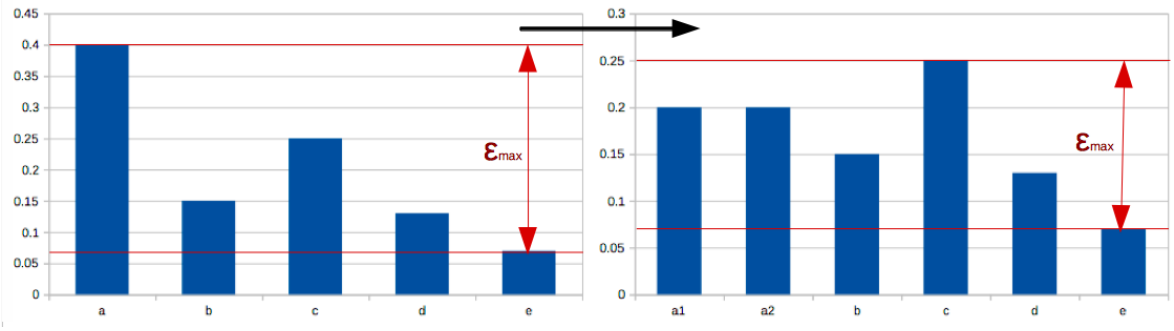
Homofónna substitúcia môže byť skonštruovaná rôznymi spôsobmi, od ktorých závisí aj jej bezpečnosť. Predpokladáme, že úspešné zlomenie šifry s vyššou bezpečnosťou znamená zlomenie aj šifier s nižšou mierou bezpečnosti. Preto je potrebné sa zamerať na lúštenie takých inštancií, ktoré predstavujú vyššiu mieru obtiažnosti.

Ako bolo spomenuté, hlavným prínosom šifrovania pomocou homofónnej substitúcie je vyhladenosť frekvenčnej charakteristiky písmen v ZT, ktorá sa v prípade monoalfabetickej substitúcie zachováva. Rovnomerné rozloženie písmen v texte závisí od konštrukcie šifry, ktorú môžeme opísať dvoma jednoduchými krokmi: vytvorenie kľúča (množina homofónov $\mathcal{H}$ s veľkosťami $\mathcal{L}$) a postupné nahrádzanie znakov OT znakmi z množín homofónov $H_i \in \mathcal{H}$ (proces šifrovania). Zložitosť lúštenia homofónnej substitúcie je teda okrem dĺžky vstupného textu ovplyvnené aj konštrukciou kľúča (počtu pridaných homofónov) a spôsobom šifrovania. Z toho vyplýva, že nevhodný návrh kľúča alebo zlý postup šifrovania môže viesť k značnému oslabeniu bezpečnosti šifry.

Na dosiahnutie rovnomerného rozdelenia znakov v ZT je dôležité správne skonštruovať samotný kľúč šifry. Pri vytváraní kľúča sa najprv pridelí ku každému znaku z $\mathcal{A}_P$ prislúchajúca množina H_i, počiatočnou veľkosťou $l_i = 1$, ktorá obsahuje práve jeden symbol. V tomto kroku je šifra totožná s monoalfabetickou substitúciou. Pridanie každého nového symbolu do $\mathcal{A}_c$ (ktorý sa taktiež zaradí do jednej skupiny H_i) vedie k tomu, že postupne vzniká homofónna substitúcia. Najefektívnejším spôsobom dosiahnutia vyhladenosti homofónnej substitúcie je rozdelenie najfrekventovanejšieho symbolu na dve časti, čiže minimalizácia odchýlky (ϵ) medzi symbolmi s najväčšou a najmenšou frekvenciou. Tento proces je znázornený na ukážkovom príklade na obrázku 5.2. Na horizontálnej osi X sú symboly kľúča, na vertikálnej osi sú uvedené frekvencie symbolov.

Pri postupnom vytváraní kľúča homofónnej substitúcie môžeme vychádzať z **prvého predpokladu**, t.j. že nový symbol sa pridá do tej skupiny H_i, ktorá obsahuje symbol s najväčšou frekvenciou. Na vytvorenie takého kľúča môžeme použiť Algoritmus[1] 15, kde sa vypočítajú veľkosti množín homofónov $\mathcal{L}$. Na základe týchto veľkostí sa následne vytvoria množiny H_i.

[1]Nahradili sme Algoritmus 14, v prípade ktorého vzniká problém so zaokrúhlením.



Obr. 5.2: Príklad pridania nového homofónu

Pri konštrukcii kľúča môžeme vychádzať z frekvenčnej charakteristiky grafém jazyka alebo príslušného otvoreného textu. Použitie frekvenčnej charakteristiky OT je však málo pravdepodobné. Z historických listov vyplýva, že kľúče sa skoro vždy vytvárali nezávisle od správy a často sa použili na zašifrovanie viacerých textov [65, 66]. Na základe toho sme si stanovili predpoklad, že pri konštrukcii kľúča homofónnej substitúcie sa používa frekvenčná charakteristika grafém daného jazyka.

Algoritmus 15: Vytvorenie kľúča homofónnej substitúcie

Input: $\{p_1, \ldots, p_{26}\}$ - referenčná frekvencia grafém,
$h \leftarrow (|\mathcal{A}_c| - |\mathcal{A}_p|)$ - počet pridaných symbolov
Output: $\mathcal{L}$ obsahuje veľkosti množín homofónov, $|\mathcal{L}| = 26$

1: inicializácia $\mathcal{L} = \{1, \ldots, 1\}$
2: **for** $c \leftarrow 1$ **to** h **do**
3: nájdi prvé i s maximálnou hodnotou $\frac{p_i}{l_i}$
4: $l_i \leftarrow l_i + 1$
5: **end for**

Algoritmus končí po vykonaní h výpočtových cyklov. Je zrejmé, že postupným znižovaním maximálnej hodnoty $\frac{p_i}{l_i}$ minimalizujeme odchýlku ϵ.

Dôležitou časťou šifrovania je postupné aplikovanie (prideľovanie) prislúchajúcich znakov z H_i k znakom z $\mathcal{A}_P$. Ak vychádzame z faktu, že najviac rovnomerné rozdelenie symbolov je možné dosiahnuť v prípade, že sa každý znak z jednej skupiny homofónov $c_j \in H_i$ použije s rovnakou pravdepodobnosťou, môžeme stanovit' **druhý predpoklad**: početnosť dvoch symbolov z jednej skupiny $c_i, c_j \in H_i$ je v ZT približne[2] rovnaká. Najjednoduchším spôsobom docielenia rovnakého počtu znakov je cyklické použitie homofónov (znázornené na Obr. 5.13) z danej skupiny. Z tohoto predpokladu vyplývajú aj možné zraniteľnosti tejto šifry (sekcia 5.3.5).

[2]Rovnaký počet sa dá dosiahnuť jedine v prípade, že početnosť prislúchajúceho znaku v OT je násobkom l_i.

Na základe uvedených predpokladov sme si stanovili nasledovné východiskové body, ktoré sa dajú využiť pri lúštení:

- V prípade, že kľúč šifry bol skonštruovaný na základe Algoritmu 15, je možné odhadnúť veľkosti skupín homofónov. Kardinalitu $|\mathcal{A}_P|$ odhadneme podľa predpokladaného jazyka; $|\mathcal{A}_C|$ zistíme priamo zo ZT (predpokladáme, že $L = |\mathcal{A}_C|$). Z $|\mathcal{A}_P|$ a $|\mathcal{A}_C|$ určíme hodnotu h, pomocou ktorej dostaneme $\mathcal{L}$ priamo z Algoritmu 15.
- Početnosť symbolov z $\mathcal{A}_c$, ktoré tvoria jednu množinu homofónov H_i je približne rovnaká.
- Symboly z $\mathcal{A}_c$, ktoré tvoria jednu množinu homofónov H_i sa môžu cyklicky opakovať v texte.

5.3 Reprezentácia a lúštenie homofónnej substitúcie

Tak, ako v prípade monoalfabetickej substitúcie, aj v prípade lúštenia homofónnej substitúcie pomocou meta-heuristík je potrebné transformovať homofónnu substitúciu na optimalizačný problém. K úspešnému lúšteniu je potrebná analýza kvality účelových funkcií.
Na základe výsledkov z analýzy lúštenia monoalfabetickej substitúcie, sme sa rozhodli vo všetkých našich experimentoch používať výlučne M_2 a tie účelové funkcie, ktoré sme určili za najlepšie (Manhattan a JSD). Experimenty sme vykonali pre $n = \{30, 40, 50, 60\}$ homofónov. Všetky testy boli realizované na rozličných dĺžkach testovacej vzorky. Použili sme 100 rôznych textov pre každú zvolenú dĺžku.

5.3.1 Špecifikácia homofónnej substitúcie ako optimalizačného problému

Pri transformácii homofónnej substitúcie na optimalizačný problém postupujeme podľa všeobecnej metodiky[3] uvedenej v sekcii 4.1, ktorú musíme doplniť o reprezentáciu v závislosti od konkrétneho kryptosystému. Pri vytváraní reprezentácie homofónnej substitúcie môžeme postupovať nasledovne:

- Vytvorenie reprezentácie bez znalosti informácií ohľadne konštrukcie šifry ($HS1$).
- Vytvorenie reprezentácie s využitím informácií o konštrukcii šifry ($HS2$).

Spoločné črty pre $HS1$ a $HS2$:

- $\mathcal{A}_C$ je číselná reprezentácia použitých symbolov - Z_n, kde n je počet použitých symbolov.

[3]Používame aj označenia definované v sekcii 4.2.1.

- $\mathcal{A}_P$ je množina grafém štandardnej anglickej abecedy bez medzery - $\{a, b, \ldots, z\}$.
- Funkcia $\mathcal{F}$ ohodnocuje text dešifrovaný pomocou k, bližšie uvedené v sekcii 4.1.2.

Vytvorenie reprezentácie bez znalosti informácií ohľadne konštrukcie šifry

V prípade, že neberieme do úvahy zistenia ohľadne konštrukcie šifry zo sekcie 5.2, najintuitívnejšie je vytvorenie reprezentácie, kde kľúčom je n prvková množina ľubovoľných znakov abecedy[4]. V tomto prípade každý znak z $\mathcal{A}_C$ môže predstavovať ľubovoľný znak z $\mathcal{A}_P$. Táto voľba však prináša isté nedostatky. Priestor možných riešení obsahuje veľké množstvo málo pravdepodobných riešení. V prípade nižšieho počtu homofónov je jasné, že je kontraproduktívne sa zaoberať skúmaním riešení, ktoré pozostávajú napr. z väčšieho počtu málo frekventovaných písmen abecedy, alebo len z rovnakých písmen abecedy.

Reprezentácia homofónnej substitúcie $HS1$:

- Θ je množina všetkých reťazcov z $\mathcal{A}_C$ dĺžky n.
- Riešenie $\theta = (x_1, x_2, \ldots, x_n)$, kde $x_i \in \mathcal{A}_P$.
- Susedné riešenia $\mathcal{N}(\theta)$ sú všetky možné zmeny[5] každého x_i v θ.

Tabuľa 5.1 obsahuje unicity distance homofónnej substitúcie (pre rôzne počty homofónov) na základe voľby modelu jazyka. Priestor kľúčov je množina všetkých reťazcov z $\mathcal{A}_C$ dĺžky n (26^n). Z Tabuľky 5.1 vidíme, že minimálnu dĺžku testovacej vzorky je nutné upresniť na základe $|\mathcal{A}_C|$.

r	1	2	3	1	2	3	1	2	3	1	2	3
n	30	30	30	40	40	40	50	50	50	60	60	60
U	282	141	101	376	188	135	470	235	168	564	282	202

Tabuľka 5.1: Unicity distance homofónnej substitúcie (reprezentácia $HS1$) pre rôzne rády modelov r

Vytvorenie reprezentácie s využitím informácií o konštrukcii šifry

Ako bolo vyššie spomenuté, môžeme uvažovať aj o prípadoch, kedy máme bližšiu informáciu o štruktúre kľúča. V prípade, že sa kľúč šifry vytváral pomocou postupu zo sekcie 5.2, môžeme na základe $|\mathcal{A}_c|$ odhadnúť veľkosti jednotlivých skupín homofónov, ako aj presnejšie určiť koľko krát sa dané písmeno abecedy vyskytuje v kľúči. Následne θ môžeme reprezentovať pomocou permutačného problému.
Reprezentácia homofónnej substitúcie $HS2$ je nasledovná:

[4]Táto reprezentácia sa uvádza v skoro každej dostupnej literatúre z tejto oblasti.
[5]$|\mathcal{N}(\theta)| = \binom{n}{1}26$, po odrátaní n aktuálnych hodnôt ostáva $n26 - n$ susedných riešení.

- Množina $S = \{x_1, x_2, \ldots, x_n\}$, $x_i \in \mathcal{A}_P$, kde početnosť rovnakých[6] znakov $x_i \in S$ je l_i (predpokladáme znalosť l_i).
- Θ je množina všetkých permutácií na S.
- Riešenie $\theta \in \Theta$ je jedna permutácia na množine S.
- Susedné riešenia $\mathcal{N}(\theta)$ sú všetky permutácie získané pomocou výmeny dvoch prvkov v kľúči.

Tabuľa 5.2 obsahuje unicity distance homofónnej substitúcie (pre rôzne počty homofónov) na základe voľby modelu jazyka. V tomto prípade uvažujeme o znalosti štruktúry homofónov, čiže veľkosti množín homofónov $\mathcal{L}$. Priestor kľúčov je teda permutácia n znakov ($n!$).

r	1	2	3	1	2	3	1	2	3	1	2	3
n	30	30	30	40	40	40	50	50	50	60	60	60
U	216	108	77	319	160	114	429	215	154	545	273	195

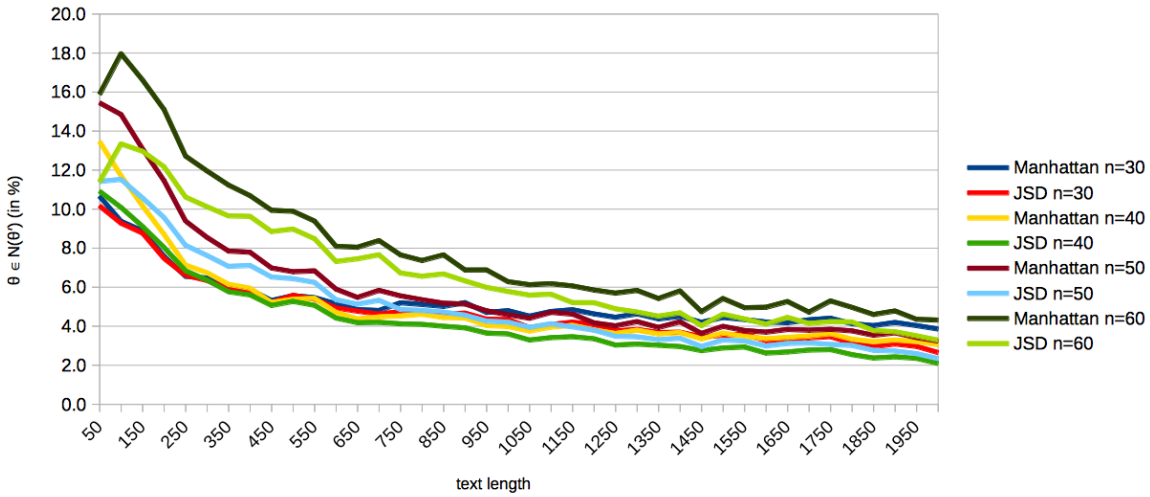
Tabuľka 5.2: Unicity distance homofónnej substitúcie (reprezentácia $HS2$) pre rôzne rády modelov r

5.3.2 Experimentálne overenie kvality účelových funkcií

Po stanovení reprezentácie homofónnej substitúcie sme sa zamerali na overenie kvality účelových funkcií. Použili sme rovnakú metodiku ako v prípade monoalfabetickej substitúcie. Jediná zmena spočívala v definícií susedných riešení (podľa zvolenej reprezentácie).
Prvá časť experimentu pozostáva z overenia hodnoty $\mathbb{B}$, t.j. či $\mathcal{N}(\theta_{corr})$ obsahuje lepšie ohodnotené riešenia ako $\mathcal{F}(\theta_{corr})$. Druhá časť experimentu skúma vzdialenosť (počet iterácií - zmien v kľúči) najbližšieho extrému od θ_{corr}. Tretia časť experimentu skúma, či sa θ_{corr} nenachádza na neutrálnom úseku.

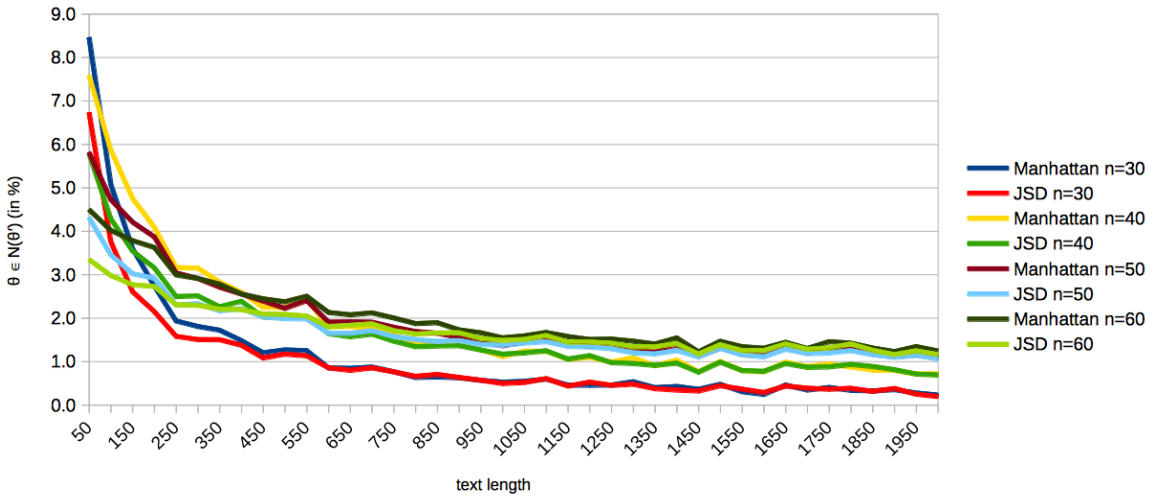
Výsledky:
V prípade homofónnej substitúcie s reprezentáciou $HS1$ a $HS2$; $\mathcal{F}(\theta_{corr}) \neq \mathcal{F}_{max}$. Počet $\mathbb{B}$ rastie v závislosti n a klesá v závislosti od dĺžky textu pre obe reprezentácie.
V prípade $HS1$ (Obr. B.42 a 5.3) tvorí počet $\mathbb{B}$ viac ako 2% susedných riešení, a to aj pre najmenšie n a najdlhšiu testovaciu vzorku. V prípade kratších textov (napr. 300 znakov) je táto hodnota v rozmedzí 6% až 12% v závislosti od hodnoty n.
V prípade $HS2$ (Obr. B.43 a 5.4) počet $\mathbb{B}$ je minimálny. Pri dĺžke textu nad 300 znakov predstavuje tento počet menej ako 3% susedných riešení a nad 900 znakov menej ako 2% (v prípade každej hodnoty n).

[6]Napr. $S = \{a, a, b, b, b, c\}$ pre $l_a = 2$, $l_b = 3$ a $l_c = 1$.

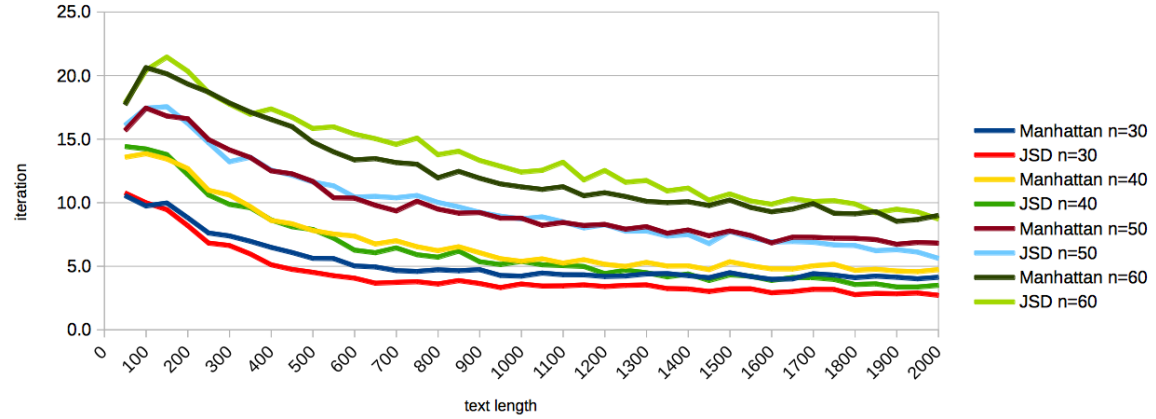


Obr. 5.3: Percento lepšie ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS1$)

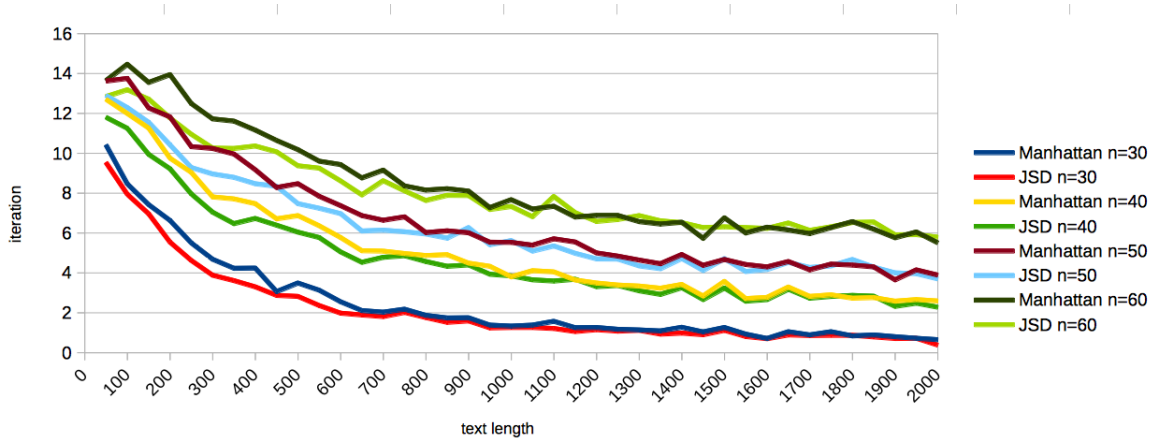
Výsledok druhej časti experimentu je vidieť na Obr. 5.5 a 5.6. V prípade oboch reprezentácií je vidieť, že vzdialenosť θ_{corr} od najbližšieho extrému rastie v závislosti od n a klesá v závislosti od dĺžky textu. Dosiahnuté výsledky pre konkrétnu hodnotu n sú porovnateľné pre obe účelové funkcie a to v prípade reprezentácie $HS1$ aj $HS2$. Pri voľbe $HS2$ však dosahujeme zhruba polovičnú vzdialenosť najbližšieho extrému od θ_{corr}, ako v prípade reprezentácie $HS1$. Výsledky reprezentácie $HS2$ (hlavne v prípade menších n) sú podobné výsledkom z analýzy monoalfabetickej substitúcie.



Obr. 5.4: Percento lepšie ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS2$)



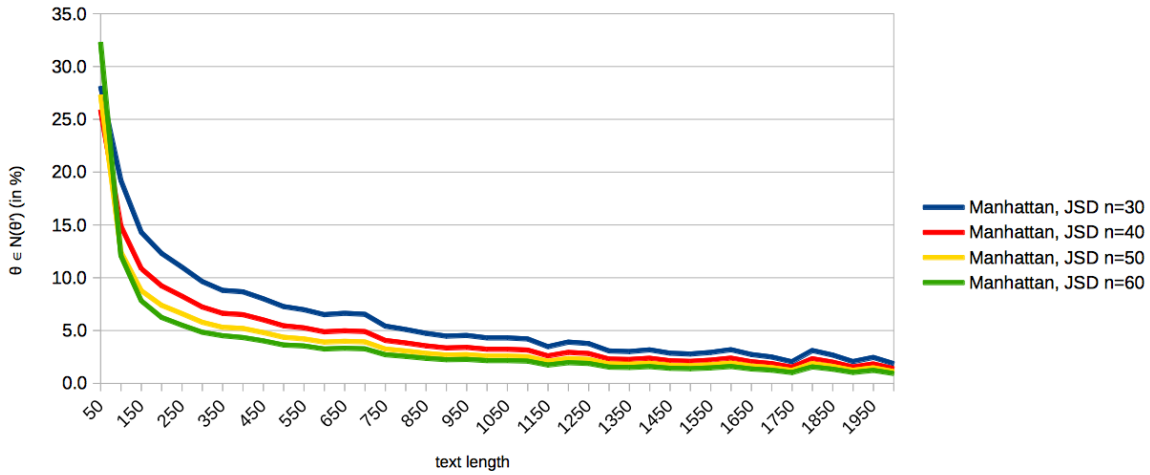
Obr. 5.5: Vzdialenosť najbližšieho extrému s lepšou fitness hodnotou od správneho riešenia pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS1$)



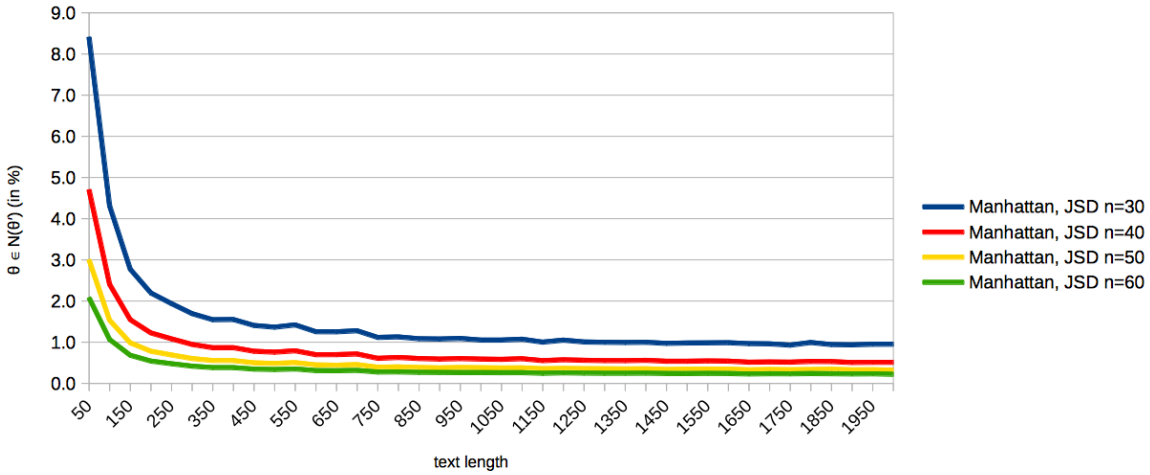
Obr. 5.6: Vzdialenosť najbližšieho extrému s lepšou fitness hodnotou od správneho riešenia pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS2$)

Výsledok tretej časti experimentu je vidieť na Obr. 5.7 a 5.8. Počet $\theta \in \mathcal{N}(\theta_{corr})$ s rovnakou hodnotou $\mathcal{F}$ s rastúcou dĺžkou textu prudko klesá a to v prípade oboch reprezentácií a nezávisí od voľby $\mathcal{F}$. Zaujímavosťou je, že od dĺžky textu cca. 200 znakov veľkosť neutrálneho úseku nezávisí (je približne rovnaký vid'. Obr. B.44 a B.45) ani od hodnoty n. Rozdiel v Obr. 5.7 a 5.8 je spôsobený tým, že veľkosť $|\mathcal{N}|$ rastie v závislosti od hodnoty n.

Najmenšie neutrálne úseky dosahujeme v prípade reprezentácie $HS2$, kde od dĺžky 200 znakov je veľkosť neutrálnych úsekov menšia ako 2% susedných riešení, podobne ako v prípade monoalfabetickej substitúcie.



Obr. 5.7: Percento rovnako ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS1$)



Obr. 5.8: Percento rovnako ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS2$)

Z výsledkov experimentu môže konštatovať':

- Nájdenie θ_{corr} pri lúštení bude závisieť hlavne od hodnoty n, dĺžky testovacej vzorky a od zvolenej reprezentácie.
- Voľba $\mathcal{F}$ vplýva najmenej na dosiahnuté výsledky.
- Hodnota $\mathcal{F}(\theta_{corr}) \neq \mathcal{F}_{max}$ ani v prípade $HS1$ a $HS2$, ale nachádza sa pomerne blízko k extrémom. Táto vzdialenosť rastie v závislosti od veľkosti n.

- Veľkosť neutrálnych úsekov je minimálna a závisí hlavne od zvolenej reprezentácie.

V prípade všetkých troch častí experimentu sme najlepšie výsledky dosiahli celkovo pri reprezentácií $HS2$.

5.3.3 Lúštenie na základe zvolenej reprezentácie

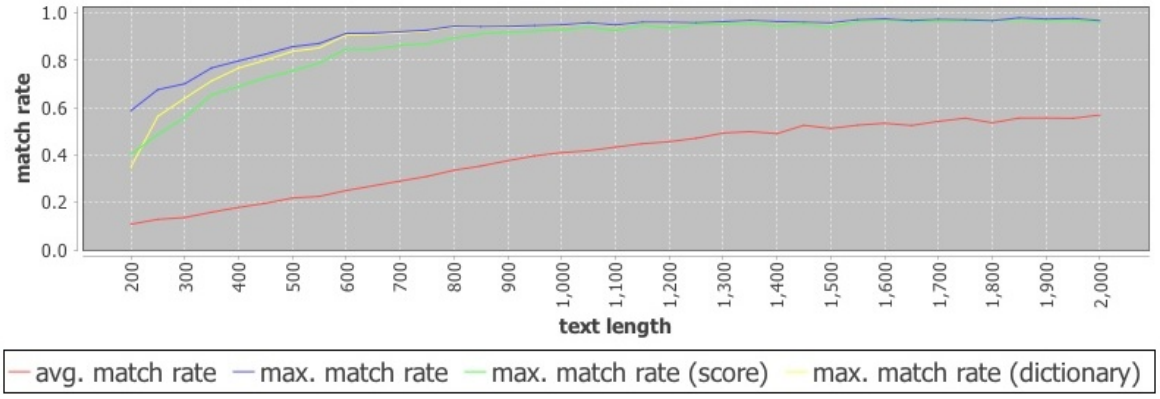
Pri lúštení sme skúmali, či je možné získať dostatočne správne riešenie, t.j. $\mathbb{M} = 80\%$ v prípade použitia reprezentácie $HS1$ a $HS2$.

Metodiku lúštenia sme navrhli na základe analýzy lúštenia monoalfabetickej substitúcie (sekcia 4), kde najefektívnejšou metódou bola SAHC meta-heuristika s reinicializáciou. Z výsledkov sekcie 5.3 vyplýva, že účelové funkcie Manhattan a JSD sú veľmi podobné, preto sme sa rozhodli pri lúštení homofónnej substitúcie použiť iba funkciu Manhattan, ktorá je výpočtovo menej náročná.

Výsledky:
V prípade $HS1$ (Obr. 5.9) sme nedosiahli dostatočne dobré výsledky ani pre najmenšiu skúmanú zložitosť $n = 30$. V prípade reinicializácie priemerná hodnota $\mathbb{M} = 60\%$.

V prípade $HS2$ sme dosiahli $\mathbb{M} = 80\%$ do zložitosti $n = 50$ (Obr. B.47), vrátane. Pri najmenšej zložitosti $n = 30$ (Obr. 5.10) sme dosiahli $\mathbb{M} = 80\%$ už pri dĺžke textu 500 znakov a pri zložitosti $n = 50$ (Obr. B.47) pri dĺžke textu 2000 znakov. Pre $n = 60$ sme neboli schopní dosiahnuť $\mathbb{M}$ nad 60%.

Obr. 5.9: Priemerná hodnota $\mathbb{M}$ pre $n = 30$ (reprezentácia $HS1$)



Obr. 5.10: Priemerná hodnota $\mathbb{M}$ pre $n = 30$ (reprezentácia $HS2$)

Z výsledkov vyplýva, že reprezentácia $HS1$ nie je pri lúštení použiteľná, preto ju v ďalších častiach práce už nebudeme používať. Požadovanú hodnotu $\mathbb{M} = 80\%$ sa nám podarilo dosiahnuť jedine v prípade reprezentácie $HS2$ (a to pre $n = \{30, 40, 50\}$). Z výsledkov môžeme konštatovať, že úspešnosť lúštenia pre $HS2$ závisí od počtu použitých homofónov ako aj od dĺžky textu.

5.3.4 Využitie početnosti symbolov v skupine homofónov pri lúštení

V prípade reprezentácie $HS2$ využívame informácie, ktoré nám umožňujú odhad presnejšej štruktúry kľúča homofónnej substitúcie. Okrem určenia veľkostí množín homofónov l_i, môžeme využiť aj fakt, že jednotlivé symboly ktoré tvoria jednu skupinu homofónov H_i by mali mať približne rovnaký výskyt v texte. Túto informáciu môžeme použiť na rozšírenie účelovej funkcie na tzv. multi-kriteriálne rozhodovanie.
Pred samotným hodnotením θ môžeme ohodnotiť priamo výmenu dvoch prvkov v permutácii, čiže vykonanú zmenu v θ. V prípade výmeny dvoch prvkov $x_i, x_j \in \theta$ priraďujeme symbol x_i zo skupiny H_i do H_j, ako aj x_j zo skupiny H_j do H_i. Podľa početnosti symbolov x_i, x_j vieme negatívne ohodnotiť tie výmeny (nové priradenia písmen), ktoré narúšajú očakávané početnosti v skupine H_i alebo H_j. Namiesto negatívneho ohodnotenia však môžeme priamo obmedziť volanie účelovej funkcie v takých prípadoch, kde sa narúšajú očakávané početnosti.
Narušenie očakávanej početnosti môžeme zistiť pomocou postupu z Algoritmu 16, ktorý vráti logickú hodnotu **false** v prípade, že sa nenaruší početnosť aspoň v jednej skupine po výmene dvojice. Relatívnu frekvenciu symbolov $x_i \in \theta$ môžeme zistiť priamo zo ZT. Očakávanú frekvenciu symbolov skupiny H_i určíme pomocou referenčnej frekvencie písmena i a veľkosti l_i.

Algoritmus 16: Zistenie narušenia očakávanej početnosti skupiny homofónov

Input: $a, b \in \theta$ - dvojica symbolov priradené do skupín $a \in H_i$ a $b \in H_j$, ktoré sa idú vymieňať s relatívnou frekvenciu p_a, p_b;
e_i, e_j - očakávané frekvencie písmen i, j ktoré reprezentujú skupiny H_i a H_j;
veľkosti $l_i \leftarrow |H_i|$ a $l_j \leftarrow |H_j|$;
povolená chybovosť δ_i, δ_j.

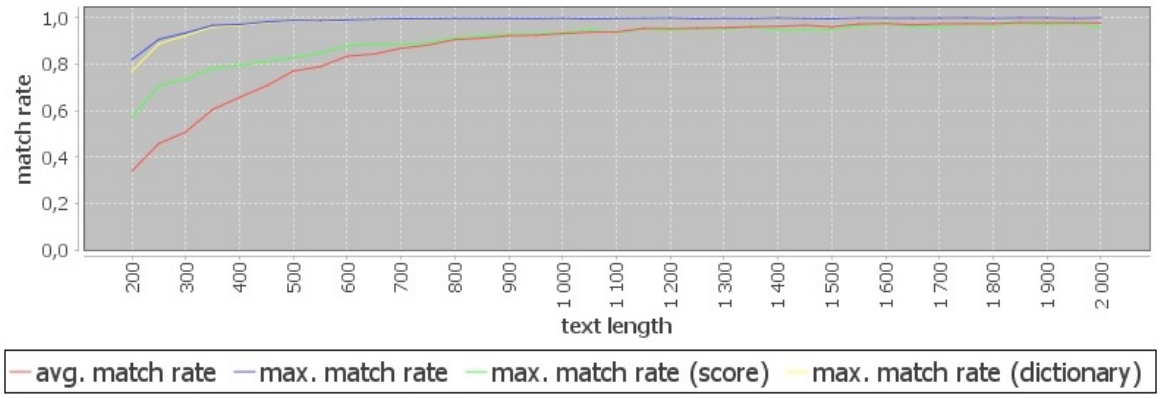
Output: (**true**) v prípade narušenia

1: **if** $|p_b - (\frac{e_i}{l_i})| < \delta_i$ **then**
2: **return false**
3: **end if**
4: **if** $|p_a - (\frac{e_j}{l_j})| < \delta_j$ **then**
5: **return false**
6: **end if**
7: **return true**

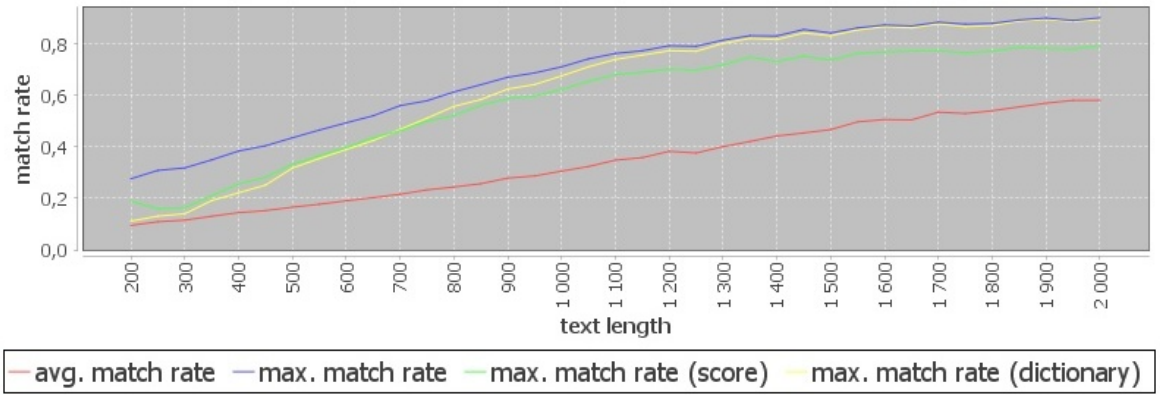
Algoritmus končí po overení dvoch podmienok a na základe ich vyhodnotenia vráti príslušnú logickú hodnotu.

V ideálnom prípade (keď frekvencie grafém v texte kopírujú očakávané frekvencie grafém jazyka) môžeme nastaviť minimálnu chybovosť δ.
Pomocou experimentu sme overili, nakoľko je možné vylepšiť úspešnosť lúštenia v ideálnom prípade, keď poznáme presnú frekvenciu grafém v texte (namiesto referenčných hodnôt e_i použijeme presnú frekvenciu písmen z OT).

Výsledky:
Z výsledkov (Obr. 5.11 a 5.12) vidíme, že dodatočné overovanie samotnej zmeny prináša jednoznačné vylepšenie úspešnosti lúštenia. Požadovanú hodnotu $\mathbb{M} = 80\%$ dosahujeme už aj v prípade zložitosti $n = 60$ od dĺžky textu 1200 znakov. Pri najmenšej zložitosti $n = 30$ dosahujeme $\mathbb{M} = 80\%$ už od dĺžky textu 200 znakov (namiesto dĺžky 500).



Obr. 5.11: Priemerná hodnota $\mathbb{M}$ pre $n = 30$ (overenie zmeny v H_i)



Obr. 5.12: Priemerná hodnota $\mathbb{M}$ pre $n = 60$ (overenie zmeny v H_i)

Treba však pripomenúť, že pri reálnom lúštení nemáme k dispozícii frekvenciu písmen OT, správne nastavenie δ_i je potrebné bližšie preskúmať.

5.3.5 Využitie cyklickej štruktúry pri lúštení

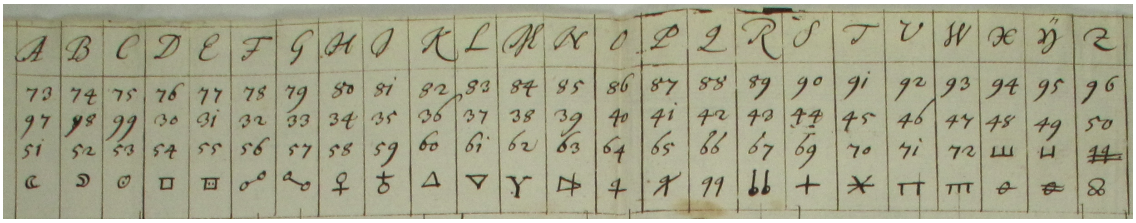
Niektoré historicky známe homofónne šifry (napr. Zodiac-ova šifra Z408) sa podarilo vylúštiť na základe nedostatkov konštrukcie šifry. Medzi najväčšie nedostatky homofónnej substitúcie patria použitie nedostatočného počtu homofónov a cyklické opakovanie symbolov v texte [3, 76, 44]. V doterajších experimentoch sme ukázali, že v prípade menšieho počtu použitých symbolov (homofónov) je možné zostrojiť dostatočne efektívny spôsob lúštenia.

V tejto časti práce skúmame možnosť identifikácie cyklicky sa opakujúcich symbolov a využitia týchto cyklov pri lúštení.

Príklad cyklického opakovania symbolov je znázornené na obrázku 5.13. Tieto cykly umožňujú identifikovať, ktoré homofóny predstavujú ten istý znak, čím sa odstráni "homofonizácia", a šifra sa približuje k monoalfabetickej substitúcii. Ďalšou slabinou môže byť použitie rovnakého množstva homofónov pre každú grafému. V tomto prípade si tabuľku homofónov môžeme predstaviť ako niekoľko riadkov abecied (napr. obrázok 5.14, a nomenklátory 1,3 a 4 z [66]). Štatistika otvoreného textu potom "kopíruje" štatistiku jednotlivých abecied. Pri dostatočne dlhom zašifrovanom texte tieto štatistiky umožňujú stotožniť niektoré najčastejšie homofóny (a dokonca stotožniť ich priamo so znakom otvoreného textu). Pri lúštení homofónnej šifry je snaha jednak o stotožnenie homofónov navzájom, a jednak o ich stotožnenie s príslušným znakom otvorenej abecedy, podobne ako pri obyčajnej substitúcii.

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

Obr. 5.13: Príklad cyklickej štruktúry šifry Z408 (prevzaté z [3])



Obr. 5.14: Časť kľúča pre homofónnu substitúciu (prevzaté z [66], nomenklátor č. 3, upravené)

Základná idea lúštenia spočíva v identifikácii cyklickej štruktúry symbolov zo ZT, na základe ktorej sa pokúsime odhadnúť príslušné symboly, ktoré tvoria jednu skupinu homofónov H_i. V prípade, že sme schopní nájsť všetky skupiny homofónov $H_i \in \mathcal{H}$, homofónnu substitúcie môžeme transformovať na monoalfabetickú.
Samotné lúštenie môžeme rozdeliť do dvoch krokov:

1. Redukcie kľúča na základe cyklickej štruktúry.
2. Lúštenie po redukcii pomocou SAHC s reinicializáciou.

Na odhad množín homofónov H_i sme navrhli metódu, ktorá využíva grafový algoritmus. Symboly z $\mathcal{A}_c$ reprezentujú vrcholy v grafe, hrany vyjadrujú cyklickú štruktúru

symbolov. Prvým krokom je určenie všetkých hrán pomocou Algoritmu 17. Cieľom je nájsť vrcholy, ktoré sú navzájom spojené hranou (tvoria kliku), t.j. vykazujú cyklickú štruktúru. Po určení hrán grafu sa použije Bron–Kerboschov algoritmus [7] na nájdenie klík, ktoré sa postupne uložia do premennej C.
Množiny v C zoradíme zostupne na základe ich veľkosti. Zoberieme prvú množinu C_0 z C a odstránime všetky ostatné množiny $C_i \in C$, ktoré obsahujú aspoň jeden symbol $y \in C_0$. Všetky symboly $y \in C_0$ v ZT nahradíme jedným znakom $\hat{y}$. Po zlúčení vybranej skupiny, C_0 odstránime z C . Pokračujem kým neodstránime všetky prvky z C. Po redukcii dostávame text, ktorý obsahuje 26 alebo menej symbolov.

Pri vytváraní kľúča bolo potrebné použiť postup zo sekcie 5.2 a počas šifrovania cyklicky striedať symboly z jednej skupiny homofónov H_i.

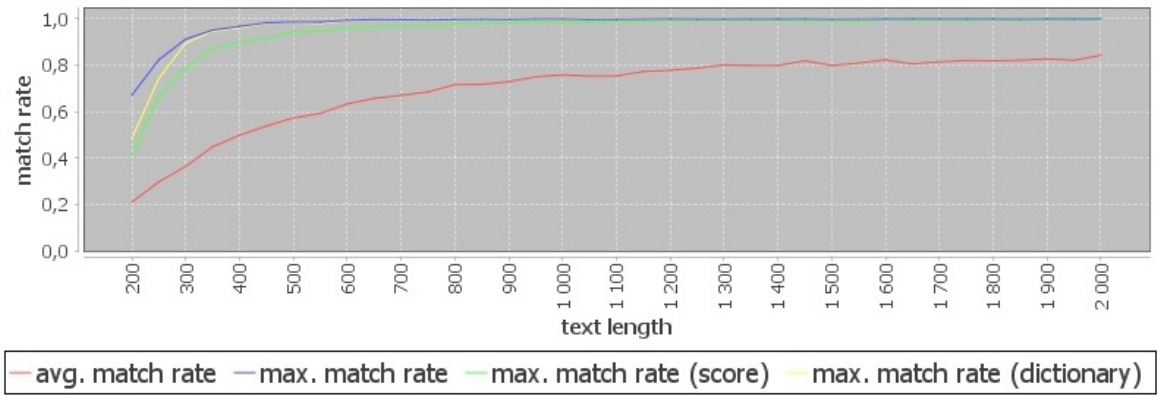
Algoritmus 17: Nájdenie dvojice symbolov s cyklickou štruktúrou

Input: i, j - dvojica symbolov;
ZT - vstupný text;
$minimalCount$ - minimálny výskyt i, j v ZT;
$maxErr$ - maximálny počet chýb pri cyklickom opakovaní i, j.
Output: (**true**) v prípade, že dvojica i, j vykazuje cyklickú štruktúru

```
1: S ← pozície znakov i, j v ZT v poradí podľa výskytu
2: if |S| < minimalCount then
3:    return false
4: end if
5: prev ← S[0]
6: err ← 0
7: for i ← 1 to |S| do
8:    if S[i] = prev then
9:       err ← err + 1
10:   end if
11:   prev ← S[i]
12: end for
13: if err > maxErr then
14:   return false
15: end if
16: return true
```

Algoritmus končí po vykonaní konečného počtu krokov: získanie množiny S trvá $|ZT|$ krokov; *for* cyklus v bode 7 sa vykoná $|S|$-krát, ktorej veľkosť je maximálne $|ZT|$. Výsledok je logická hodnota podľa splnenia uvedených jednoduchých podmienok.

Výsledky:
V uvedených experimentoch, v prípade zachovania bezchybnej cyklickej štruktúry sme vždy boli schopní nájsť správne skupiny homofónov a redukovať text na maximálne 26 znakov. V niektorých prípadoch sme však našli a spojili menej frekventované znaky, ktoré netvorili jednu množinu H, ale napriek tomu vykazovali cyklickú štruktúru. Táto skutočnosť však neovplyvnila zásadným spôsobom úspešnosť pri lúštení.
Z obrázku 5.15 je vidieť, že už nad dĺžkou 250/300 znakov sme schopní dosiahnuť úspešnosť $\mathbb{M} = 80\%$ v prípade počiatočných 60 homofónov. V prílohe na obrázkoch B.49, B.50 a B.51 uvádzame výsledky lúštenia po redukcii v prípade 30, 40, 50 homofónov.



Obr. 5.15: Priemerná hodnota $\mathbb{M}$ pri reinicializácií po redukcii z $n = 60$

Kapitola 6

Overenie funkčnosti vytvorenej metodiky

V kapitolách 4 a 5 sme opísali novo vytvorenú metodiku lúštenia monoalfabetickej a homofónnej substitúcie. Funkčnosť metodiky bola overená prostredníctvom lúštenia veľkého množstva nami vytvorených šifrovaných textov. V prípade monoalfabetickej substitúcie sme dosiahli priaznivé výsledky a to aj v prípade problémových krátkych šifrovaných textov. Lúštenie homofónnej substitúcie sa ukázalo byť problémové pri veľkom počte použitých symbolov (homofónov). Prijateľné výsledky lúštenia v prípade homofónnej substitúcie sme dosiahli len pri dlhých šifrovaných textoch a len pri splnení istých predpokladov.

Za dôležité pokladáme overiť kvalitu našej metodiky v prípade lúštenia šifrovaných správ z reálnych historických zdrojov. Preto sme sa rozhodli o experimentálne lúštenie niekoľkých vybraných substitučných šifier, ktoré sa zachovali v rôznych archívoch a publikáciách.

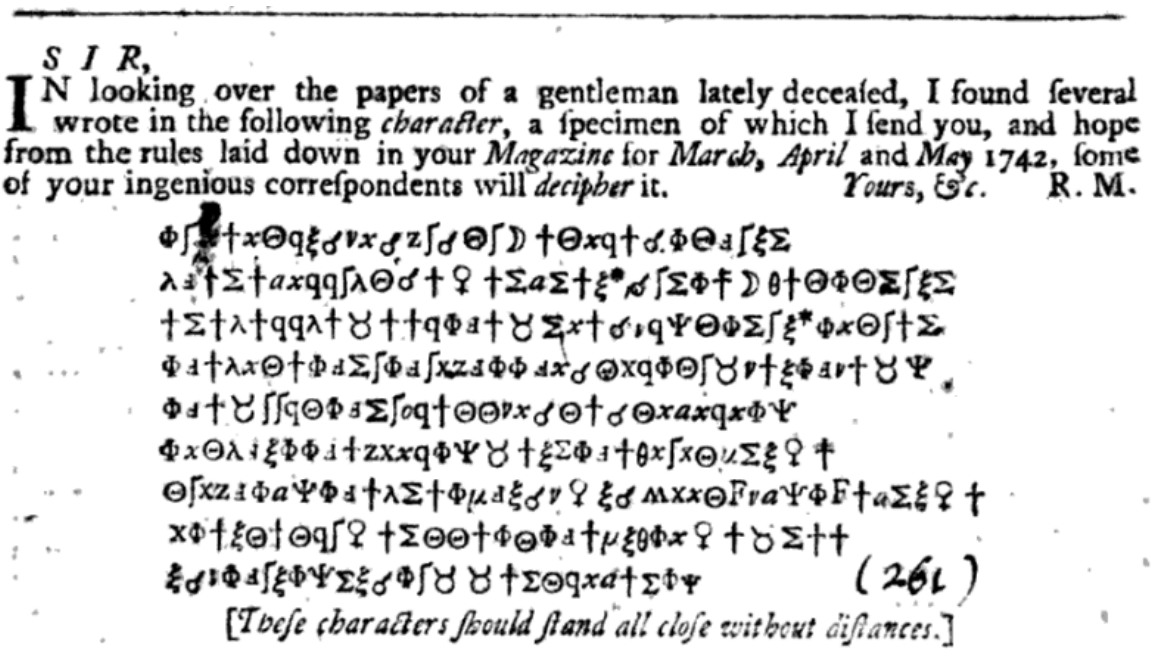
Lúštenie monoalfabetickej substitúcie sme overili pomocou troch krátkych šifrovaných správ, ktoré predstavujú zložitejšiu inštanciu problematiky. K týmto ZT existujú aj príslušné OT. Dodatočne sa nám podarilo nájsť jednu šifrovanú správu (z českých archívov), ku ktorej sme v čase lúštenia nemali k dispozícii príslušný OT. Vo všetkých prípadoch sme použili SAHC s reinicializáciou ($r = 1000$), účelovú funkciu JSD a Manhattan s modelom jazyka M2 (sekcia 4.3).

V prípade homofónnej substitúcie sme sa zamerali na známu Zodiac-ovu zlomenú šifru Z408. Najprv sme pri lúštení využili odhad štruktúry kľúča (reprezentácia HS2 podľa sekcie 5.3.3) a následne cyklickú štruktúru homofónov (sekcia 5.3.5). Použili sme SAHC s reinicializáciou ($r = 5000$), účelovú funkciu Manhattan s modelom jazyka M2.

V nasledujúcich sekciách uvádzame stručné informácie a výsledky nášho lúštenia. Znalosť OT sme nevyužili pri lúštení, len pri overení kvality výsledkov.

6.1 Lúštenie šifry Gentlemen's cipher

Prvá zvolená monoalfabetická substitúcia je známa ako *Gentlemen's cipher*, ktorá bola publikovaná v magazíne *The Gentlemen's Magazine* v roku 1748. Šifra bola zlomená až v roku 2013 Nickom Pellingom [72]. Zašifrovaný text je uvedený na obrázku 6.1. Správa je dĺžky 301 symbolov.



S I R,

IN looking over the papers of a gentleman lately deceaſed, I found ſeveral wrote in the following *character*, a ſpecimen of which I ſend you, and hope from the rules laid down in your *Magazine* for *March*, *April* and *May* 1742, ſome of your ingenious correſpondents will *decipher* it. *Yours, &c.* R. M.

(261)

[*Theſe characters ſhould ſtand all cloſe without diſtances.*]

Obr. 6.1: Gentlemen's cipher (prevzaté z [104])

Použili sme modifikovaný prepis správy v nasledovnej forme (symboli boli prevedené na čísla):

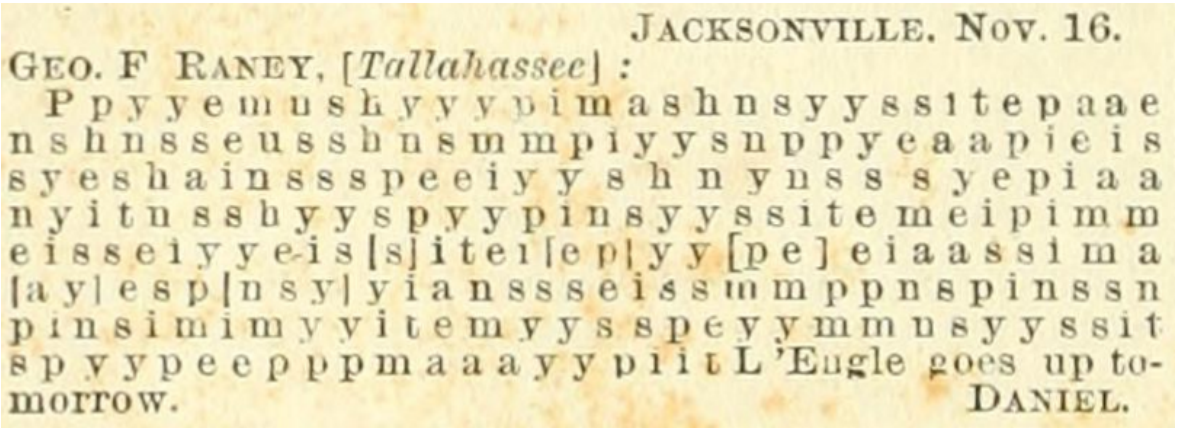
0 1 2 3 4 3 5 6 7 8 2 3 8 9 1 8 5 1 10 4 5 3 6 4 8 0 5 11 1 12 4 13 11 4 12 4 14 3 6
6 1 13 5 8 4 15 4 12 16 12 4 7 17 8 1 12 0 4 10 18 4 5 0 5 12 1 7 12 4 12 4 13 4 6 6
13 4 19 4 4 6 0 11 4 19 12 3 4 8 2 6 20 5 0 12 1 17 4 0 3 5 1 4 12 0 11 4 13 3 5 0 11
12 1 0 11 1 3 9 11 0 0 11 4 3 8 5 3 6 0 5 1 19 2 4 7 0 11 2 4 19 20 0 11 4 19 1 1 6 5
0 11 12 1 14 6 4 5 0 3 8 3 4 8 5 3 14 3 6 3 0 20 0 3 5 13 11 7 0 0 11 4 9 3 3 6 0 20
19 4 7 12 0 11 4 18 3 1 3 5 1 12 7 15 4 5 1 3 9 11 0 14 20 0 4 13 12 4 0 1 11 7 8 2
15 7 8 21 3 3 5 11 4 2 14 20 0 11 4 14 12 7 15 4 3 0 4 7 5 4 5 6 1 15 4 12 5 5 12 0 5
0 11 4 22 7 18 0 3 15 4 18 12 4 4 7 8 2 0 11 1 7 0 20 12 7 8 0 1 19 19 4 12 5 6 3 14 4 12 0 20

Správny otvorený text je dostupný v [72].

Ako najlepší výsledok sme dosiahli $\mathbb{M} = 94.35\%$ a to v prípade oboch účelových funkcií.

6.2 Lúštenie šifier z amerických prezidentských volieb z roku 1876

Nasledujúce dve monoalfabetické substitúcie sú z čias amerických prezidentských volieb z roku 1876. Jedná sa o šifrovanú komunikáciu politikov z Floridy (Secret Operation in Florida [80]). Uvedené šifrované správy pochádzajú z korešpondencie, ktorú uvádzal aj W. Friedman vo svojej učebnici [86].
Na obrázku 6.2 je prvá šifrovaná správa dĺžky 120 znakov s pracovným názvom *Daniel* (pomenované podľa odosielateľa). Na obrázku 6.3 je šifrovaná správa dĺžky 272 znakov s pracovným názvom *Engle*, kde každé dvojciferné číslo reprezentuje jeden znak.


JACKSONVILLE. NOV. 16.

GEO. F RANEY, [*Tallahassee*] :

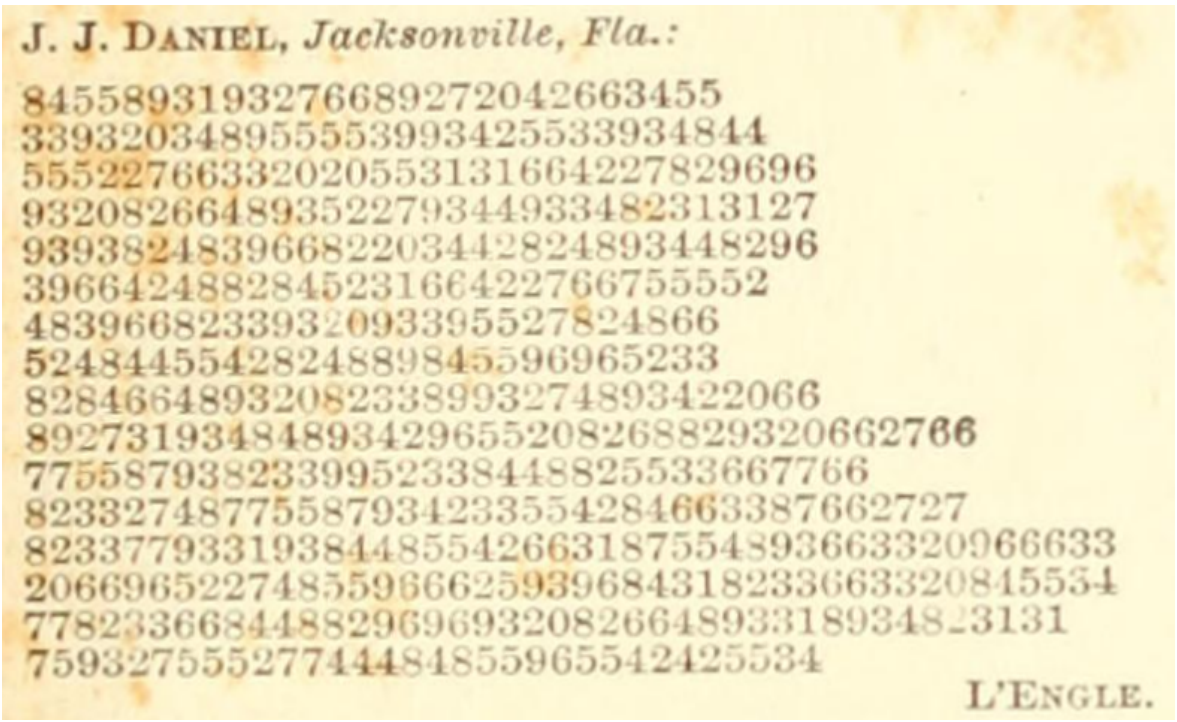
P p y y e m u s h y y y p i m a s h n s y y s s i t e p a a e
n s h n s s e u s s h n s m m p i y y s n p p y e a a p i e i s
s y e s h a i n s s s p e e i y y s h n y n s s s y e p i a a
n y i t n s s h y y s p y y p i n s y y s s i t e m e i p i m m
e i s s e i y y e i s [s] i t e i [e p] y y [p e] e i a a s s i m a
[a y] e s p [n s y] y i a n s s s e i s s m m p p n s p i n s s n
p i n s i m i m y y i t e m y y s s p e y y m m n s y y s s i t
s p y y p e e p p p m a a a y y p i i t L'Engle goes up to-
morrow. DANIEL.

Obr. 6.2: Šifrovaný text Daniel, Secret Operation in Florida (prevzaté z [80])

Prepis správy Daniel (písmená boli prevedené na čísla):
0 1 2 3 4 1 5 6 7 3 1 8 9 10 11 12 7 3 13 14 7 3 15 5 1 16 0 17 11 5 18 8 17 7 19 3 8 20 18 1 7
21 3 8 17 5 11 21 9 3 7 1 22 1 5 3 1 8 9 2 18 5 15 18 8 18 1 18 8 9 18 10 1 20 18 11 8 23 11
17 22 3 1 24 3 8 18 8 15 0 3 5 3 16 5 3 23 23 1 9 2 1 8 20 1 25 3 1 8 9 22 1 20 10 0 6 11 1 5 9

Prepis správy Engle (dvojice cifier boli prevedené na čísla, podľa poradia prvého výskytu):
0 1 2 3 4 5 6 2 5 7 8 6 9 1 10 4 7 9 2 1 1 11 4 8 1 10 4 12 13 1 14 5 6 10 7 7 1 3 3 6 8 5 15 16
16 4 7 15 6 12 4 14 5 4 13 4 9 15 3 3 5 4 4 15 12 11 6 15 7 9 8 15 12 4 13 15 16 11 6 8 12 15
0 14 3 6 8 5 6 17 1 14 12 11 6 15 10 4 7 4 11 1 5 15 12 6 14 12 13 1 8 15 12 2 0 1 16 16 14
10 15 0 6 12 4 7 15 10 2 4 5 12 4 8 7 6 2 5 3 4 12 12 4 8 16 1 7 15 18 15 4 7 6 5 6 19 1 20 4
15 10 21 14 10 0 12 15 1 10 6 19 6 15 10 5 12 19 1 20 4 8 10 1 8 0 6 10 20 6 5 5 15 10 19 4
3 4 0 12 1 8 6 3 20 1 12 4 6 10 7 16 6 10 7 6 16 14 5 12 1 16 6 22 4 16 0 3 15 10 6 10 7 0 1
9 19 15 10 6 0 12 15 16 16 4 7 15 6 12 4 3 2 9 15 3 3 17 4 5 1 14 19 13 12 12 1 16 1 8 8 1 9

Správne otvorené texty sú dostupné v [80].


J. J. DANIEL, *Jacksonville, Fla.:*

8455893193276892720426634 55
3393203489555539934255339348 44
5552276633202055313166422782 9696
9320826648935227934493348231 3127
9393824839668220344282489344 8296
3966424882845231664227667555 52
4839668233932093395527824866
5248445542824889845596965233
8284664893208233899327489342 2066
8927319348489342965520826882 9320662766
7755879382339952338448825533667766
8233274877558793423355428466338766 2727
8233779331938448554266318755489366332 0966633
2066965227485596662593968431823366332 0845534
7782336684488296969320826648933189348 3131
7593275552774448485596554242 5534

L'ENGLE.

Obr. 6.3: Šifrovaný text Engel, Secret Operation in Florida (prevzaté z [80])

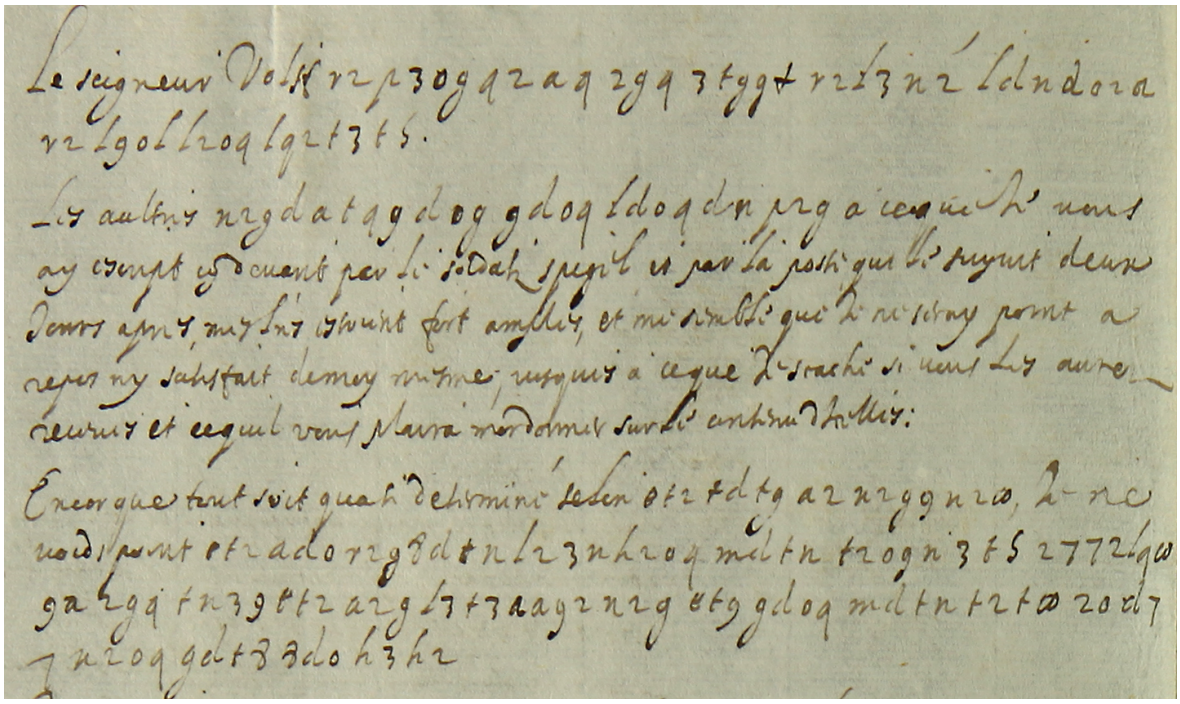
Ako najlepší výsledok v prípade šifry Daniel sme dosiahli $\mathbb{M} = 76.6\%$ v prípade účelovej funkcie JSD a $\mathbb{M} = 71\%$ v prípade účelovej funkcie Manhattan.

V prípade šifry Engle sme dosiahli $\mathbb{M} = 94.4\%$ v prípade účelovej funkcie JSD a až $\mathbb{M} = 96.6\%$ v prípade účelovej funkcie Manhattan.

6.3 Lúštenie šifrovaných správ z českých archívov

Zaujímavým zdrojom šifrovaných správ sú české [65, 66] ako aj slovenské historické archívy. Pomocou Mgr. Jakuba Mírku sa nám podarilo získať šifrované správy z českých archívov [97] z obdobia tridsaťročnej vojny (ukážka na obrázku 6.4, originálna správa priložená v prílohe A), konkrétne z 3. a 5. marca 1619. Autorom správy je agent Jaquot, recipientom je Karel Bonaventura Buquoy [107].

Počas lúštenia sme nemali k dispozícii prislúchajúci OT ani kľúč. Po vylúštení správy sa nám podarilo získať rukopis z archívu, ktorý obsahuje prepis listu z 3. marca 1619, v dešifrovanej podobe (príloha A), čím sa podarilo potvrdiť úspešnosť lúštenia.



Obr. 6.4: Ukážka šifrovaného textu z českých archívov [97]

Jazyk zašifrovanej časti sme predpokladali na základe jazyka častí textu, ktoré neboli šifrované (text je písaný po francúzsky). Číselný prepis šifrovanej časti sme vykonali ručne (príloha A), ktorý celkovo tvorí 2603 znakov. Po vykonaní frekvenčnej analýzy (ako aj na základe počtu použitých symbolov - 24 rôznych) sme odhadli, že na šifrovanie sa použila monoalfabetická substitúcia. Na získanie referenčných štatistík sme použili francúzsky korpus z [57].

Najlepší výsledok sme dostali pomocou funkcie JSD, ktorý uvádzame aj v prílohe A. V prípade funkcie Manhattan sme dosiahli len 67%-nú zhodu grafém s najlepším výsledkom JSD. Po analýze výsledku vidíme, že text je síce písaný po francúzsky, ale vykazuje isté odlišnosti od bežnej formy francúzštiny (napr. nerozlišovali písmená *u* a *v*).
Nasledujú vybrané pasáže[1] po dešifrovaní s prekladom do slovenského jazyka:

- iehan de nassau aura part a ses troppes - Johan de Nassou sa bude podieľať na tomto vojsku;
- sortie des hongrois - útok (únik) Maďarov;
- cinc cent hommes de garde au chasteau de prague - 500 mužov na stráži Pražského hradu;

[1] V rozdelení textu na slová a pri preklade nám pomohla RNDr. Karla Čipková, PhD.

- iehan de Nassau au rapart a ses troppes - Johan de Nassou sa bude podieľat' na tomto vojsku;
- l ambassadeur a receu cent mille escus et asseurance de cinc cent mille - veľvyslanec dostal sto tisíc korún a zábezbeku vo výške päťsto tisíc;
- pour romper et chastier l ennemy - aby rozbil a kruto potrestal nepriateľa;
- quinze mille florins - pätnásť tisíc zlatých.

6.4 Lúštenie Zodiac-ovej šifry Z408

Pseudonym Zodiac patrí jednému z najznámejších sériových vrahov z Kalifornie, ktorý poslal viacero zašifrovaných správ (ďalej Zodiac-ove šifry) do rôznych novín v roku 1969. Prvú šifru, Z408 zlomili ručne v pomerne krátkom čase po jej publikovaní. Druhú šifru, Z340 sa však však do dnešného dna nepodarilo rozlúštiť. Kvôli veľkému počtu použitých symbolov a malej dĺžky šifrovaných správ patria tieto šifry k zložitejším inštanciám homofónnej substitúcie.

Zamerali sme sa na lúštenie šifry Z408 (dĺžka textu je 408; počet symbolov je 54). Kľúč šifrovanej správy je na obrázku 5.1, číselný prepis na obrázku 5.13. V prípade využitia štruktúry kľúča (reprezentácia HS2) sme dosiahli $\mathbb{M} = 31.02\%$. Napriek tomu, že štruktúra kľúča v prípade Z408 nezodpovedá úplne našim predpokladom (Tabuľka 6.1), výsledok lúštenia zodpovedá očakávaným hodnotám.

Znak	l podľa Alg. 15	l v Z408	rozdiel	Znak	l podľa Alg. 15	l v Z408	rozdiel
a	4	5	1	d	2	2	0
e	8	7	1	h	2	2	0
i	4	6	2	l	2	3	1
n	4	4	0	o	4	5	1
r	2	3	1	s	3	4	1
t	4	2	0				

Tabuľka 6.1: Porovnanie počtu symbolov očakávanej homofónnej substitúcie a Z408

V druhej časti lúštenia sme sa zamerali na využitie cyklickej štruktúry homofónov (Z408 vykazuje len čiastočne cyklickú štruktúru symbolov). Po redukcii na 33 symbolov sme dosiahli $\mathbb{M} = 41.79\%$. Pri zvýšení dovolenej chybovosti sme boli schopní redukovať počet symbolov na 26 s výsledkom $\mathbb{M} = 48.2\%$. Výsledky s uvedenou hodnotou $\mathbb{M}$ však nie sú čitateľné.

Záver

Využitie moderných optimalizačných metód - meta-heuristík predstavuje aktuálny trend vo výskume kryptoanalýzy klasických šifier. Dôležitú časť problematiky tvorí najmä transformácia kryptoanalýzy na optimalizačný problém a voľba vhodných účelových funkcií (s využitím analýzy textu). Jedným z najväčších nedostatkov tejto oblasti je absencia ucelenej a systematickej metodiky, čo veľakrát prispieva k nevhodnej adaptácii meta-heuristík. S tým súvisí napríklad aj nízka úspešnosť pri lúštení krátkych šifrovaných správ, alebo pri lúštení homofónnej substitúcie.
Teoretická časť predkladanej práce obsahuje zhrnutie aktuálnych poznatkov z troch kľúčových oblastí: kryptoanalýza klasických šifier, meta-heuristiky a analýza textu. Na základe zistených nedostatkov súčasného stavu problematiky sme si stanovili nasledovné ciele dizertačnej práce:

- Vytvorenie ucelenej metodiky kryptoanalýzy klasických šifier pomocou meta-heuristík.

- Zhodnotenie možnosti efektívneho využitia meta-heuristík pri lúštení monoalfabetickej substitúcie.

- Vytvorenie nových postupov lúštenia homofónnej substitúcie.

Prvému a druhému cieľu sme sa venovali v kapitole 4. Najprv sme vytvorili ucelenú metodiku kryptoanalýzy s využitím meta-heuristík, ktorú sme demonštrovali pomocou lúštenia monoalfabetickej susbtitúcie. Na základe analýzy vybraných parametrov lúštenia (voľba modelu jazyka, voľba meta-heuristiky a voľba účelovej funkcie) a skúmania charakteru globálnej geometrie účelových funkcií, sme vytvorili odporúčania na dosiahnutie čo najlepších výsledkov a to aj v prípade lúštenia problematických krátkych textov. Najdôležitejšie zistenia a odporúčania sú nasledovné:

- Úspešnosť lúštenia monoalfabetickej substitúcie závisí hlavne od voľby modelu jazyka a od voľby účelovej funckie. Najvhodnejšia voľba je model jazyka založený na frekvencii 2-gramov (model M_2), a funkcie Manhatanská vzdialenosť (Manhattan) alebo Jensen-Shannonova divergencia (JSD).

- V prípade odporúčaného modelu M_2 globálny extrém účelových funkcií síce nepredstavuje správne riešenie, avšak nachádza sa v jeho blízkosti.

- V prípade modelu M_2 účelové funkcie Manhattan a JSD neobsahujú veľké neutrálne úseky (viacero možných riešení s rovnakým ohodnotením), obsahujú však väčší počet lokálnych extrémov v prípade kratších správ (do dĺžky 1000 znakov). Na riešenie tohto problému odporúčame využiť prírodou inšpirované metaheuristiky DPSO, FFA alebo SAHC s reinicializáciou. Dôležité je pri tom dodatočné ohodnotenie výsledkov pomocou slovníka.

Najlepšie výsledky lúštenia sa nám podarilo dosiahnuť pomocou SAHC s reinicializáciou. Priemernú zhodu grafém výsledku lúštenia so správnym riešením sme zvýšili už v prípade textov dĺžky 200 znakov na viac ako 90%, oproti doterajšiemu najlepšiemu výsledku 74% z [105]. V porovnaní s výsledkami z [105] to znamená dosiahnutie plne čitateľného textu.

Lúšteniu homofónnej substitúcie - čo bolo našim tretím cieľom - sme sa venovali v kapitole 5, v ktorej sme bližšie popísali vlastnosti homofónnej substitúcie. Ďalej sme skúmali možnosti vytvorenia jej reprezentácie (súčasť transformácie kryptoanalýzy na optimalizačný problém) s využitím aj bez využitia poznatkov vyplývajúcich z konštrukcie šifry.
Pomocou odhadu štruktúry kľúča šifry sme vytvorili reprezentáciu (permutačný problém), vďaka ktorej sme boli schopní lúštiť homofónnu substitúciu až do počtu homofónov $n = |\mathcal{A}_c| = 50$, vrátane. Pri najmenšom skúmanom počte homofónov $n = 30$ sme dosiahli 80%-nú zhodu grafém so správnym výsledkom už od dĺžky textu 500 znakov. Pri $n = 40$ homofónov sme dosiahli rovnakú zhodu grafém so správnym výsledkom od dĺžky textu 1200 znakov a pri $n = 50$ od dĺžky textu 2000 znakov. Pre $n = 60$ homofónov sme neboli schopní dosiahnuť čitateľný text. V prípade menšieho počtu homofónov ($n = 30, 40$) sme našimi novými metódami dosiahli výsledky porovnateľné s doteraz najlepšími dosiahnutými výsledkami z fundamentálnej práce [17]. V prípade použitia väčšieho počtu homofónov ($n = 50, 60$) však už dosahujeme horšie výsledky.
Ďalej sme ukázali, že v prípade rovnakej početnosti symbolov z príslušnej skupiny homofónov je možné rozšíriť metodiku o dodatočné ohodnotenie. Pomocou tohto ohodnotenia sa dá lúštiť aj homofónna substitúcia so zložitosťou $n = 60$ homofónov (s 80%-nou zhodou grafém so správnym výsledkom už pri textoch dĺžky 1200 znakov). Dosiahnuté výsledky boli porovnateľne lepšie ako v [17] a to aj v prípade najväčšej skúmanej zložitosti.
V prípade špeciálnej konštrukcie šifry (keď sa všetky symboly cyklicky striedajú z príslušnej skupiny homofónov) sme vytvorili postup, pomocou ktorého je možné homofónnu substitúciu redukovať na monoalfabetickú a následne využiť metodiku a odporúčania z kapitoly 4. Hľadanie homofónov s cyklickou štruktúrou sme transformovali na problém hľadania klík v grafe, využili sme pri tom Bron–Kerboschov algoritmus. V tomto prípade sme dosiahli 90%-nú zhodu grafém so správnym výsledkom už aj v prípade veľmi krátkych textov, t.j. textov dĺžky 300 znakov pri zložitosti $n = 60$ homofónov. Podobnú úspešnosť v [17] dosiahli len pri textoch dlhších ako 3000 znakov v prípade nižšej zložitosti $n = 55$ homofónov (nevyužívali analýzu cyklickej štruktúry homofónov). V prípade textov dĺžky 300 znakov dosiahli len okolo 10%-nú zhodu gra-

fém so správnym výsledkom.

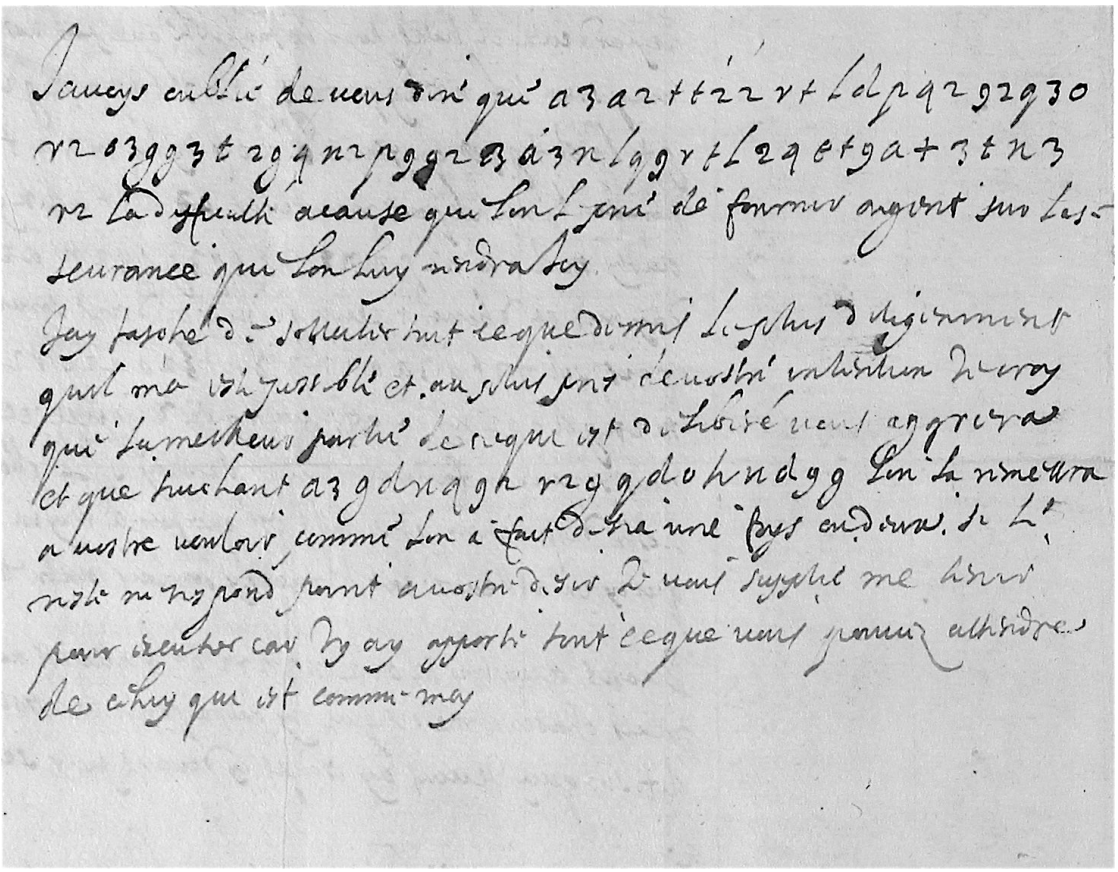
Správne fungovanie nami vytvorenej metodiky sme overili lúštením reálnych šifier získaných z archívov a z rôznych publikácií, konkrétne: Gentlemen's cipher, dve šifrované správy z amerických prezidentských volieb z roku 1876, jedna šifrovaná správa z tridsať-ročnej vojny a Zodiac-ova šifra Z408. Úspešnosť lúštenia monoalfabetických substitúcií v prípade textov dĺžky aspoň 200 znakov, kde sme mali možnosť overiť správnosť lúštenia bola viac ako 90%-ná. Pri správe, kde sme nemali k dispozícii otvorený text, sme dosiahli plne čitateľný text. Úspešnosť lúštenia homofónnej substitúcie (šifry Z408) bola porovnateľná s očakávaným výsledkom danej zložitosti ($n = 54$, dĺžka textu 408 znakov). Zhodu grafém výsledku lúštenia so správnym riešením sa nám podarilo zvýšiť na 48.2% využitím cyklickej štruktúry homofónov.

V uvedených výsledkoch sme ukázali, akým postupom je možné dosiahnuť priaznivé výsledky pri lúštení monoalfabetickej a homofónnej substitúcie. Problémom však ostáva lúštenie homofónnej substitúcie, v ktorej nie je možné využiť cyklickú štruktúru homofónov. V takomto prípade je lúštenie krátkych správ (do dĺžky 1000 znakov) problémové a to hlavne v prípade väčšieho počtu homofónov ($n = 40, 50, 60$).

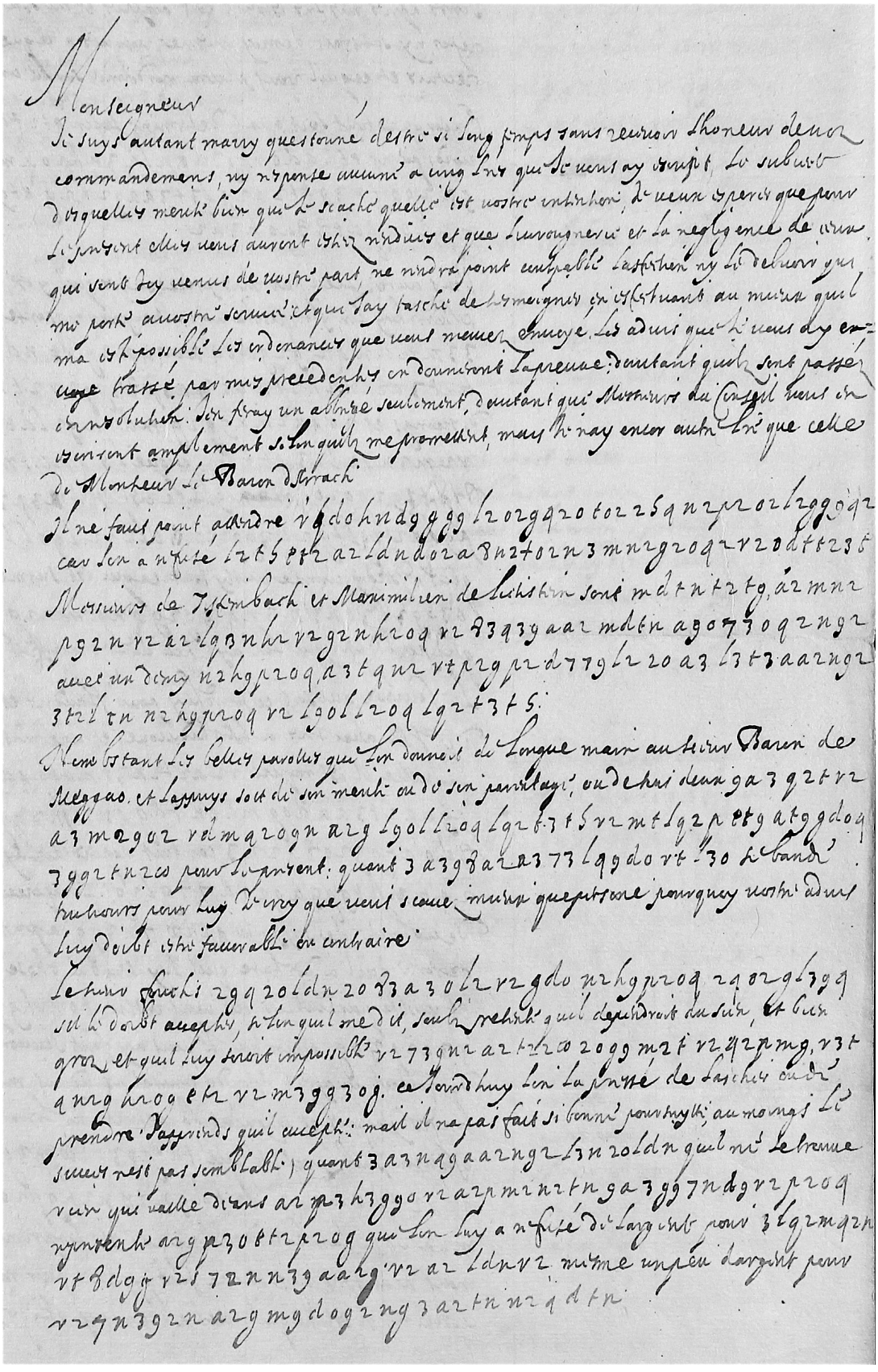
Za hlavný prínos práce považujeme nami vytvorenú ucelenú metodiku lúštenia klasických šifier pomocou optimalizačných algoritmov. Táto metodika, spoločne s vykonanými experimentmi, je zdrojom nových informácií potrebných pre vhodné použitie meta-heuristík pri lúštení. Zistené poznatky nám umožnili zvýšenie úspešnosti lúštenia problematických krátkych textov oproti doterajším prácam a sú perspektívnym podkladom aj pre ďalší rozvoj výskumu.

Dodatok A

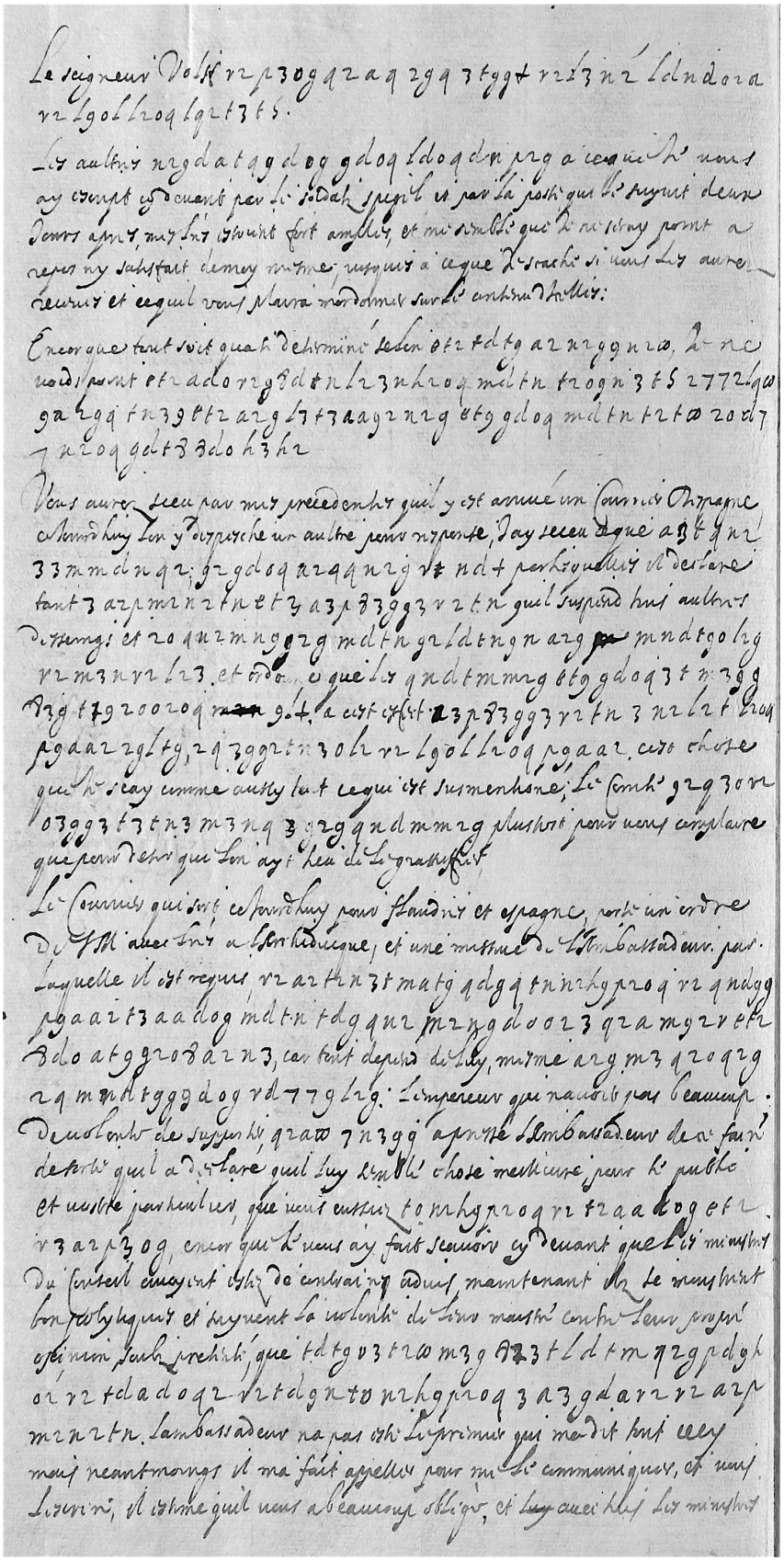
List agenta Jaquota grófovi Buquoyovi z 3. a 5. marca 1619



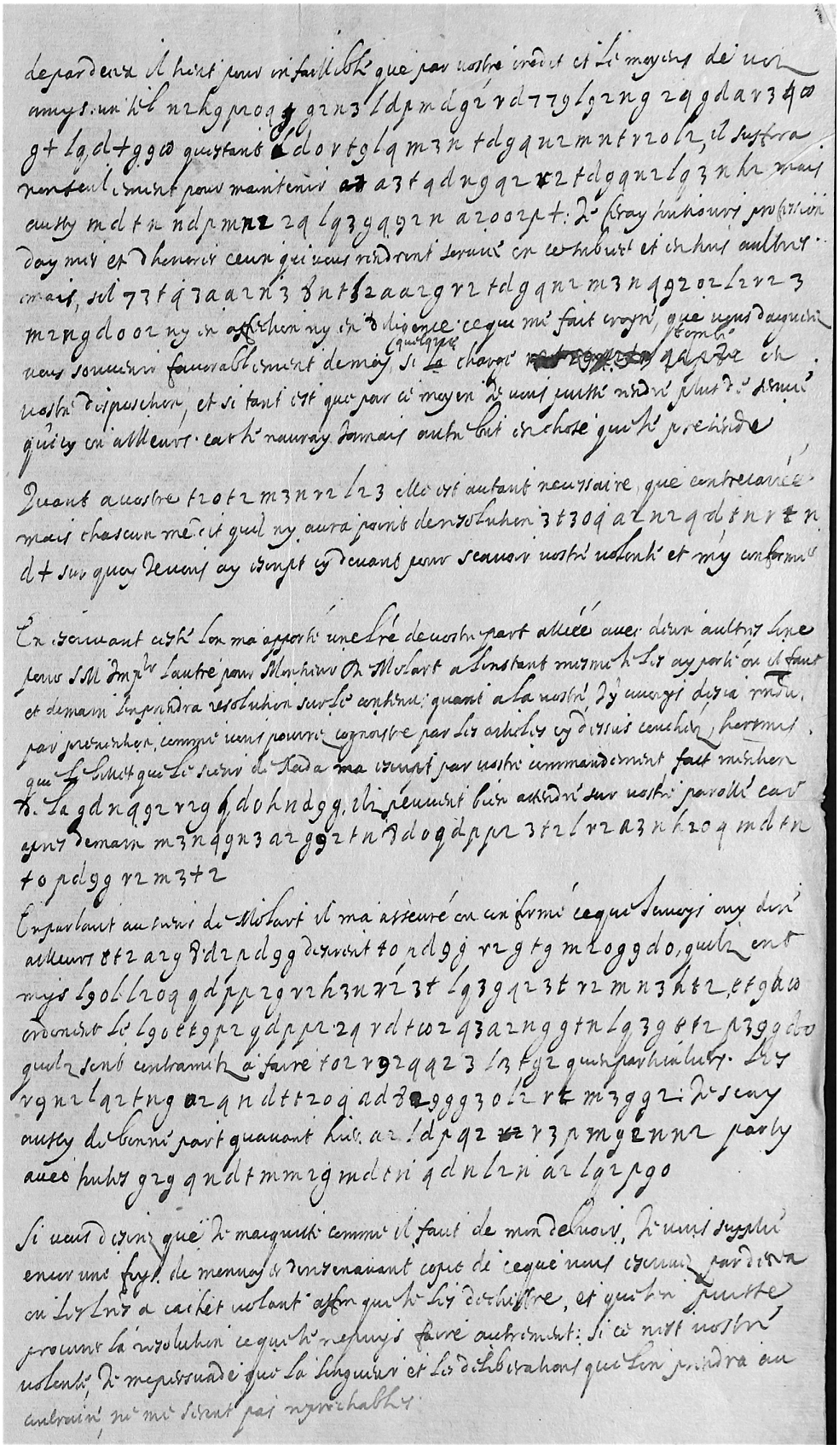
Obr. A.1: List agenta Jaquota grófovi Buquoyovi, 4. časť [97]



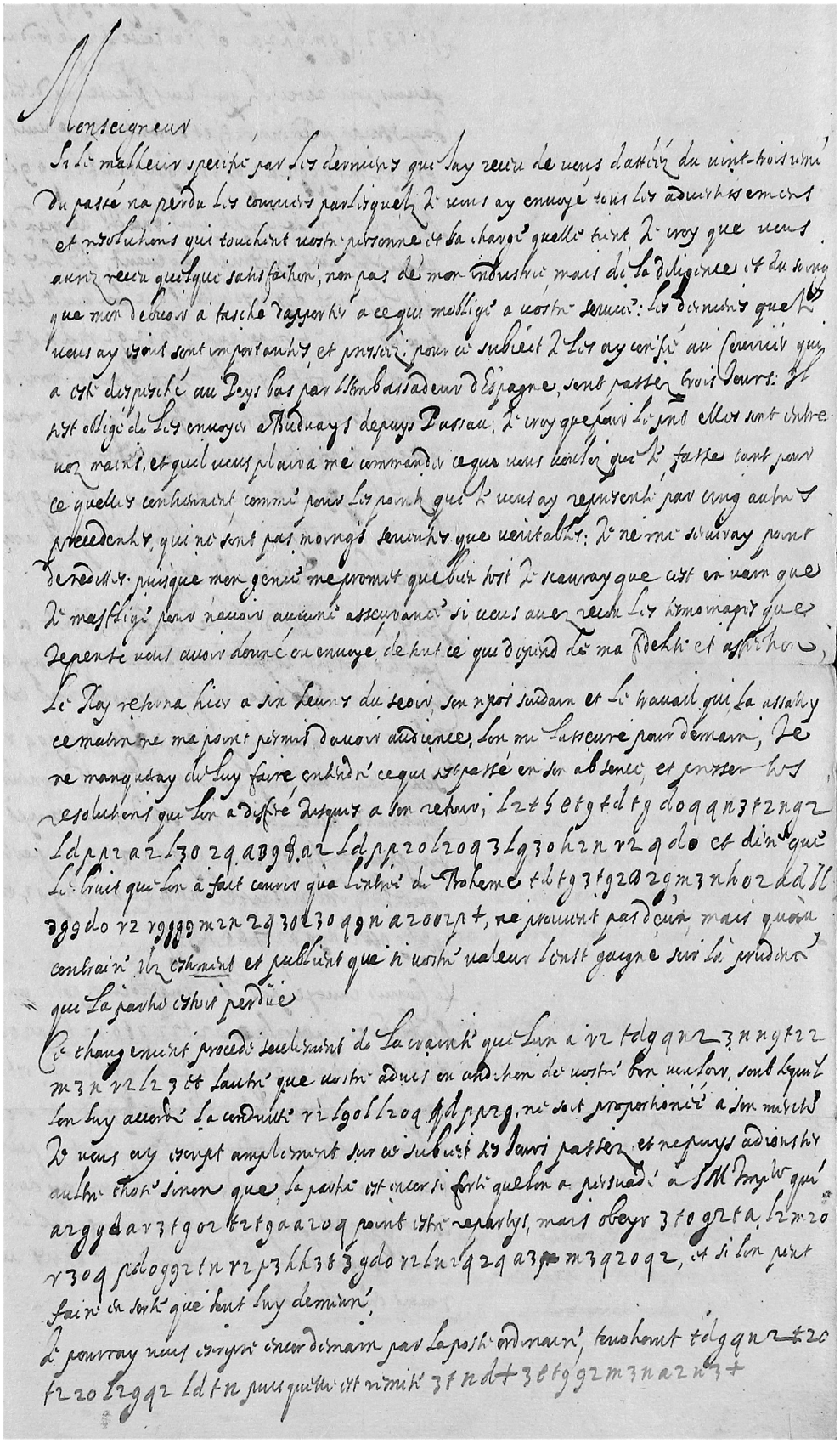
Obr. A.2: List agenta Jaquota grófovi Buquoyovi, 1. časť [97]



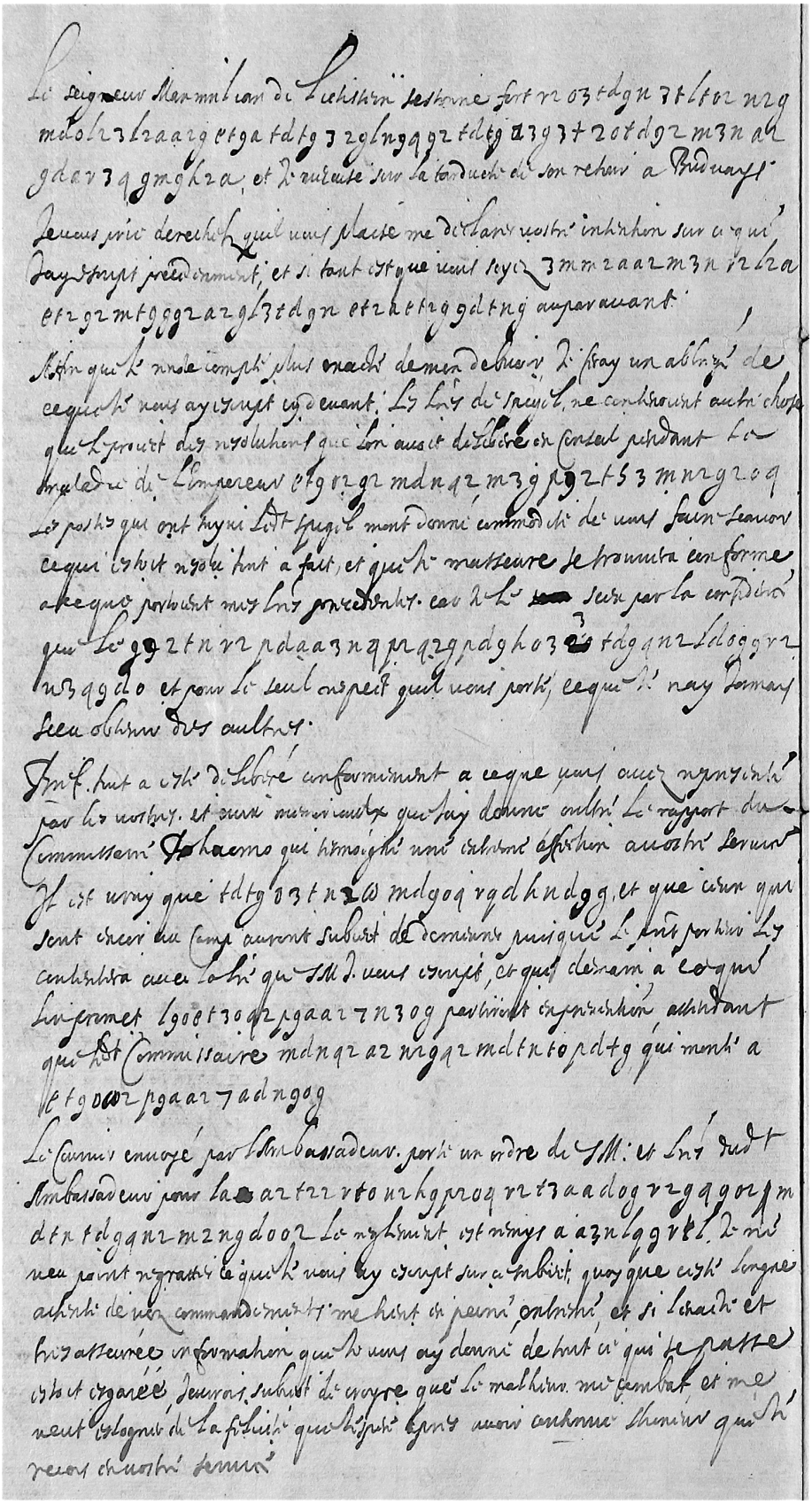
Obr. A.3: List agenta Jaquota grófovi Buquoyovi, 2. časť [97]



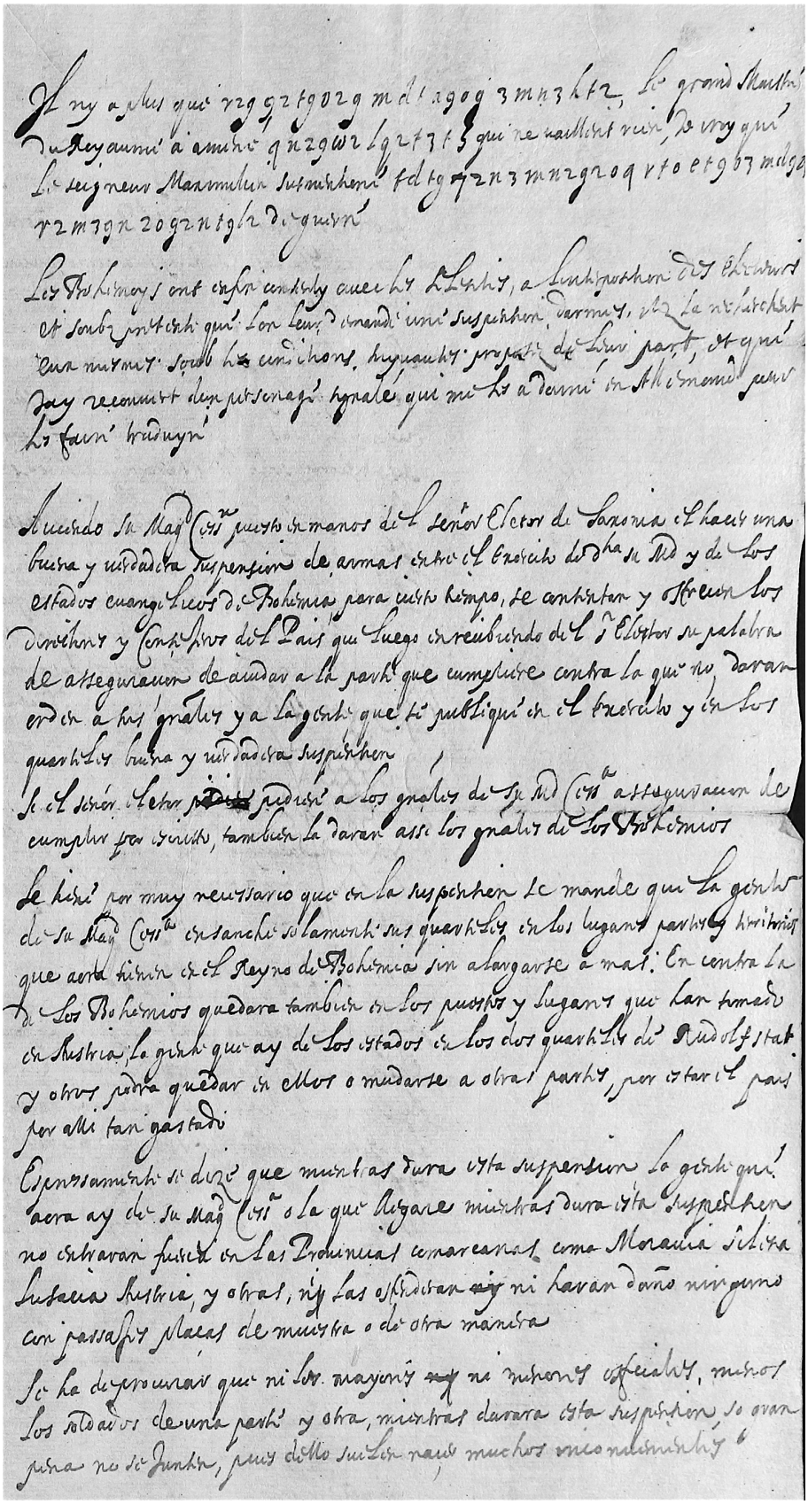
Obr. A.4: List agenta Jaquota grófovi Buquoyovi, 3. časť [97]



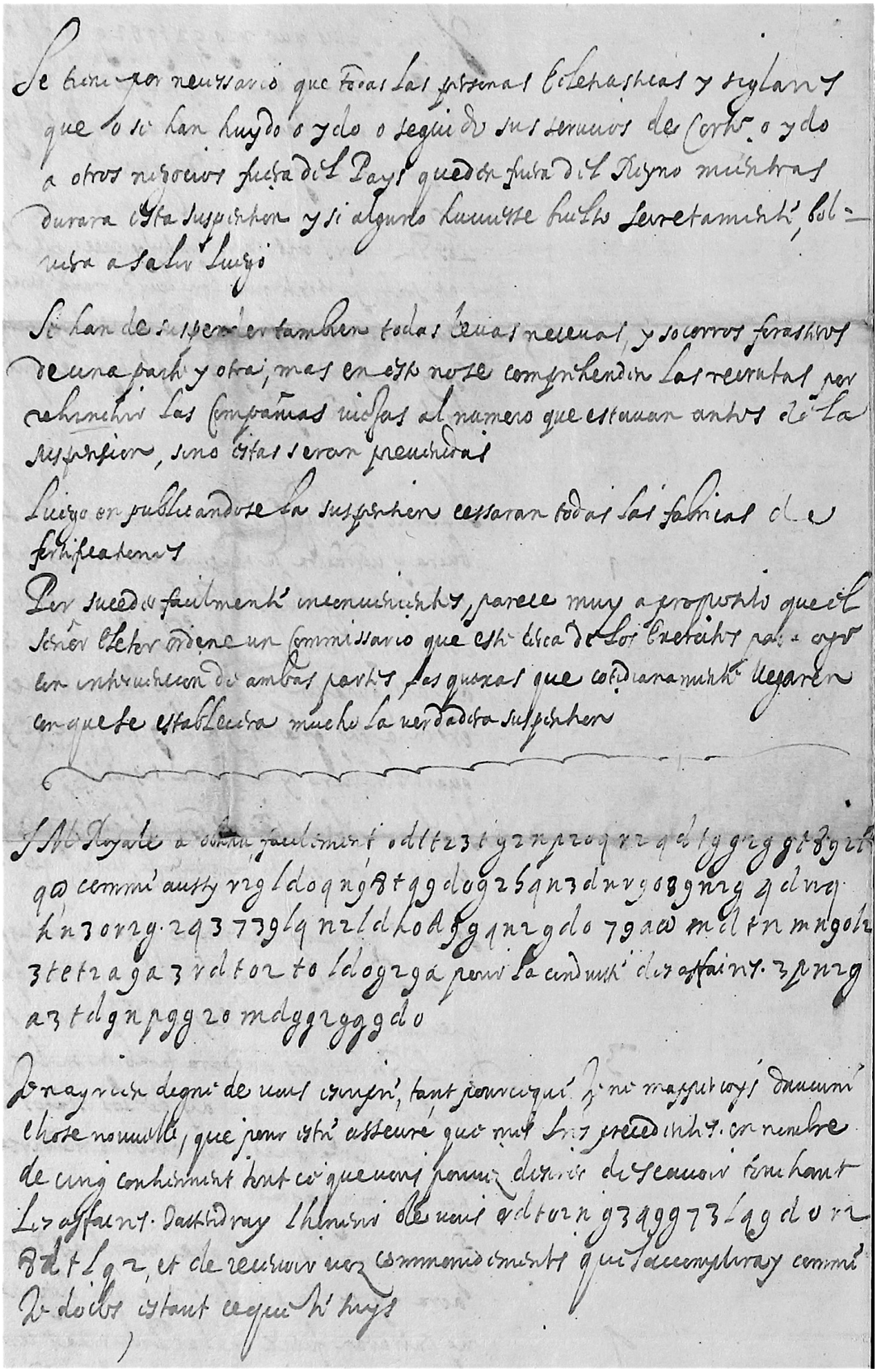
Obr. A.5: List agenta Jaquota grófovi Buquoyovi, 5. časť [97]



Obr. A.6: List agenta Jaquota grófovi Buquoyovi, 6. časť [97]



Obr. A.7: List agenta Jaquota grófovi Buquoyovi, 7. časť [97]



Obr. A.8: List agenta Jaquota grófovi Buquoyovi, 8. časť [97]

Prepis šifrovaných častí

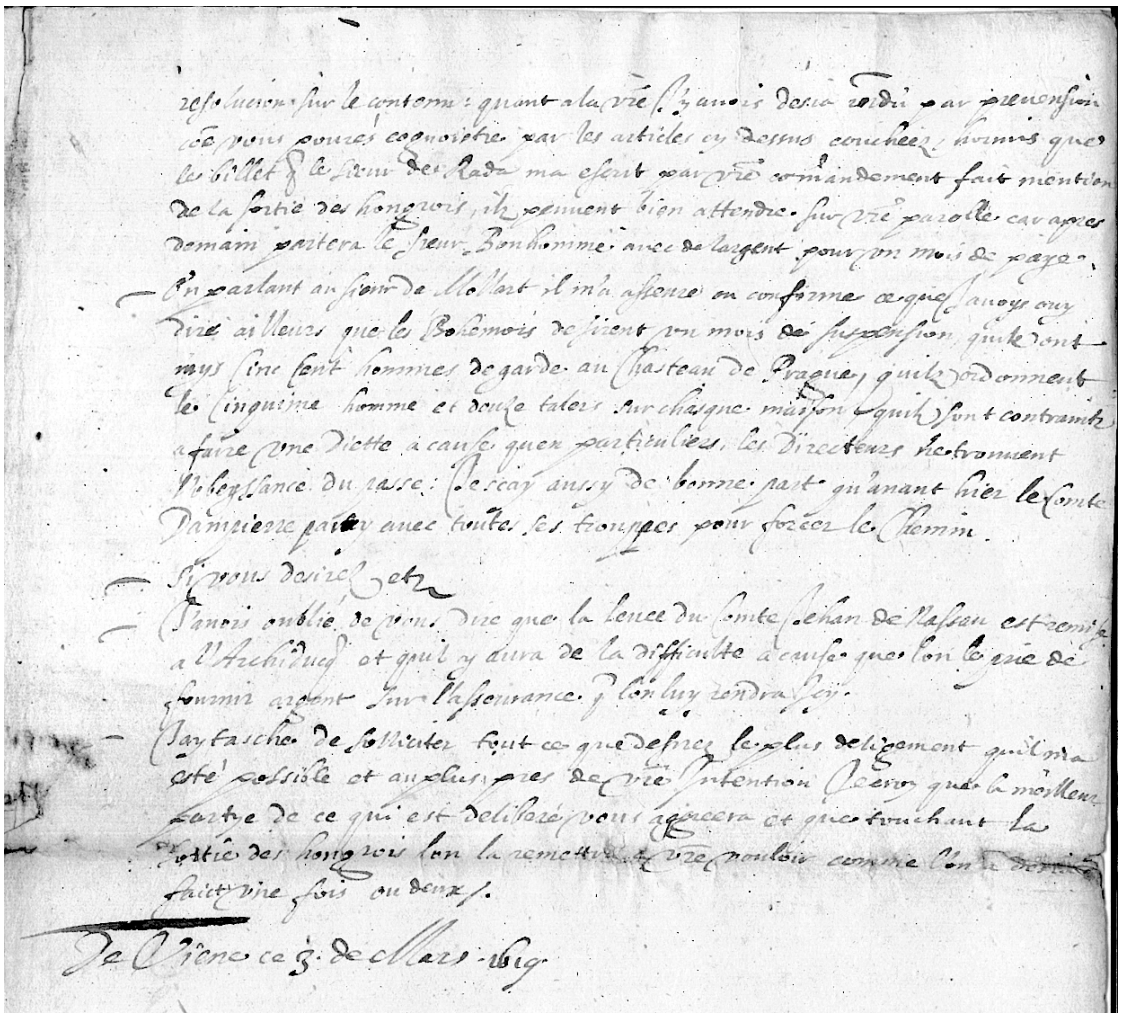
0 1 2 3 4 5 2 6 7 7 6 8 9 3 9 7 10 9 3 11 3 9 9 12 10 5 9 13 9 3 9 8 9 7 7 6 10 9 8 9 11
12 14 15 9 16 9 8 2 5 2 3 9 16 17 5 9 18 3 9 5 19 20 5 9 7 9 3 10 9 0 9 3 2 11 15 9 19
11 20 2 11 5 11 9 11 7 16 9 20 5 9 13 6 9 5 0 9 16 9 8 1 19 5 4 9 0 9 7 9 5 4 9 3 10 0 9
17 19 10 19 6 16 16 9 9 20 2 11 5 16 6 3 21 19 3 10 9 5 6 9 5 9 4 6 13 9 3 10 16 19 11
10 5 9 0 11 13 9 7 13 9 2 21 21 6 8 9 9 3 16 19 8 19 11 19 16 16 9 5 6 9 19 11 9 8 11 5
5 9 4 6 13 9 3 10 0 9 8 6 3 8 8 9 3 10 8 1 9 11 19 11 12 6 16 19 1 9 15 0 9 16 19 20 9 6
3 9 0 2 20 10 9 3 6 5 16 9 7 8 6 3 8 8 9 3 10 8 1 9 15 19 11 12 0 9 20 11 8 1 9 13 14 11
6 16 11 6 7 2 3 10 19 7 7 9 11 5 9 22 19 16 19 6 17 16 9 16 19 21 19 8 10 6 2 3 0 11 8
19 3 9 7 10 9 3 8 2 5 9 3 17 19 16 19 3 8 9 0 9 7 2 3 5 9 4 6 13 9 3 10 9 10 3 9 7 8 19 6
10 0 9 21 19 6 5 9 16 9 11 9 9 22 9 3 7 6 20 9 11 0 9 10 9 13 20 7 0 19 11 10 5 9 7 4 9
3 7 14 11 9 0 9 20 19 6 7 19 3 7 19 16 19 5 16 6 16 16 9 5 7 9 8 19 9 3 8 2 5 16 9 13 19
4 19 7 6 3 0 9 16 9 13 20 9 5 9 11 5 6 16 19 7 6 21 5 2 6 0 9 13 9 3 10 16 9 7 13 19 3
14 11 9 13 9 3 7 19 8 1 9 20 10 9 5 0 15 17 2 6 7 0 9 23 21 9 5 5 19 6 16 16 9 7 0 9 16
9 8 2 5 0 9 0 9 21 5 19 6 10 9 5 16 9 7 20 6 2 3 6 9 5 7 19 16 9 11 5 5 9 10 2 11 5 0 9
13 19 3 7 10 9 16 10 9 7 10 19 11 7 7 18 0 9 8 19 5 9 8 2 5 2 3 9 16 0 9 8 6 3 8 8 9 3 10
8 1 9 11 19 11 12 5 9 7 11 16 11 10 6 2 3 7 7 2 3 10 8 2 3 10 2 5 13 9 7 14 11 9 11 2 11
7 16 9 5 9 7 6 5 9 22 14 11 9 16 2 3 0 9 7 17 2 11 5 8 9 19 5 4 9 3 10 20 2 11 5 11 9 3 6
5 19 15 12 9 21 21 9 8 10 22 6 16 9 7 10 11 5 19 6 14 11 9 16 9 7 8 19 11 19 16 16 6 9
5 9 7 14 11 6 7 2 3 10 20 2 11 5 11 9 11 22 9 3 2 21 21 5 9 3 10 7 2 11 17 17 2 3 4 19 4
9 16 19 11 10 5 9 19 19 20 20 2 5 10 9 9 6 9 7 2 3 10 16 9 10 10 5 9 7 0 11 5 2 18 19 16
9 13 20 9 5 9 11 5 14 11 19 16 19 13 17 19 7 7 19 0 9 11 5 9 3 10 5 9 20 5 6 7 9 7 20 2
11 5 9 8 2 11 5 6 5 16 9 7 20 5 2 11 3 8 9 7 0 9 20 19 5 0 9 8 9 19 10 5 2 11 20 20 9 7
14 11 6 7 2 3 10 19 11 20 19 6 7 17 19 7 11 6 9 3 3 9 3 10 16 19 13 17 19 7 7 19 0 9 11
5 19 5 9 8 9 11 8 9 3 10 13 6 16 16 9 9 7 8 11 7 9 10 19 7 7 9 11 5 19 3 8 9 0 9 8 6 3 8
8 9 3 10 13 6 16 16 9 6 9 1 19 3 0 9 3 19 7 7 19 11 19 11 5 19 20 19 5 10 19 7 9 7 10 5
2 20 20 9 7 0 9 16 9 11 9 5 19 15 20 16 11 7 10 2 7 10 11 5 5 9 4 6 13 9 3 10 0 9 10 5 2
6 7 13 6 16 16 9 11 19 16 16 2 3 7 20 2 11 5 11 2 7 10 5 9 20 2 5 7 2 3 3 9 19 10 9 16
20 6 9 0 14 11 9 17 2 3 16 11 6 7 9 3 17 16 9 5 19 16 9 7 20 19 10 9 3 10 9 7 9 10 20 5
2 11 6 7 6 2 3 7 0 2 21 21 6 8 9 7 10 9 16 22 21 5 19 6 7 11 3 5 9 4 6 13 9 3 10 0 9 11 9
16 16 2 3 7 14 11 9 0 19 16 9 13 19 3 7 11 2 11 7 0 19 11 9 22 20 19 7 17 9 19 11 8 2
11 20 10 9 7 13 2 6 4 3 9 0 9 11 2 16 2 3 10 9 0 9 11 2 6 5 11 3 5 9 4 6 13 9 3 10 19 16
19 7 2 16 0 9 0 9 16 9 13 20 9 5 9 11 5 5 9 4 6 13 9 3 10 7 9 5 19 8 2 13 20 2 7 9 0 2 21
21 6 8 6 9 5 7 9 10 7 2 16 0 19 10 22 7 18 8 1 2 18 7 6 22 8 2 3 0 11 6 8 10 20 19 5 11
2 7 10 5 9 20 5 11 0 9 3 8 9 16 19 10 2 5 6 10 9 0 9 11 2 6 10 5 9 8 3 19 5 4 9 20 2 11 5
5 2 13 20 5 9 9 10 8 1 19 7 10 6 9 5 16 9 3 3 9 13 18 21 19 11 10 19 16 16 9 5 19 17 5
11 8 9 16 16 9 7 0 9 11 2 7 10 5 9 20 19 5 10 6 9 3 9 8 9 0 9 19 20 9 5 7 2 3 3 9 11 9 3
11 9 20 19 5 0 9 8 9 19 19 11 19 3 10 16 9 5 9 10 2 11 5 0 11 5 2 18 7 2 5 10 6 9 0 9 7
1 2 3 4 5 2 6 7 20 19 5 10 6 5 19 16 9 7 6 9 11 5 17 2 3 1 2 13 13 9 19 11 9 8 0 9 16 19
5 8 9 3 10 20 2 11 5 11 3 13 2 11 7 0 9 20 19 18 9 14 11 9 16 9 7 17 2 9 13 2 6 7 11 3 9
2 6 7 0 9 7 11 7 20 9 3 7 6 2 3 8 6 3 8 8 9 3 10 1 2 13 13 9 7 0 9 4 19 5 0 9 19 11 8 1 19
7 10 9 19 11 0 9 20 5 19 4 11 9 14 11 6 16 22 8 6 3 14 11 6 13 9 1 2 13 13 9 9 10 0 2 11
22 9 10 19 16 9 5 7 7 11 5 8 1 19 7 14 11 9 13 19 6 16 2 0 11 3 9 0 6 9 10 10 9 19 8 19

11 7 9 0 6 5 9 8 10 9 11 5 7 16 9 10 5 2 11 9 3 10 16 2 17 9 6 7 7 19 3 8 9 0 11 20 19 7 7 9 16 9 8 2 13 10 9 0 19 13 20 6 9 5 5 9 7 9 7 10 5 2 11 20 20 9 7 20 2 11 5 10 2 5 8 9 5 16 9 8 1 9 13 6 3 16 19 16 9 11 11 9 9 0 11 8 2 13 10 9 6 9 1 19 3 0 9 3 19 7 7 19 11 9 7 10 5 9 13 6 7 9 19 16 19 5 8 1 6 0 11 8 9 10 14 11 6 16 18 19 11 5 19 16 19 7 2 5 10 6 9 0 9 7 1 2 3 4 5 2 6 7 8 9 11 12 14 11 6 11 2 11 7 2 3 10 10 5 19 11 9 5 7 9 8 2 13 13 9 16 9 8 19 3 9 10 16 19 6 17 16 9 8 2 13 13 9 3 8 9 3 10 19 8 1 19 3 4 9 5 0 9 10 2 3 11 2 11 7 19 11 6 9 22 9 7 20 19 11 4 3 9 16 2 8 8 19 7 6 2 3 0 9 0 6 7 7 6 20 9 5 9 10 19 3 9 19 3 10 6 5 16 9 3 3 9 13 18 0 9 11 2 7 10 5 9 19 5 5 6 11 9 9 20 19 5 0 9 8 9 19 11 15 0 9 8 6 3 8 8 9 3 10 1 2 13 13 9 7 16 9 7 7 2 16 0 19 11 7 3 9 11 9 15 6 16 16 9 3 10 19 11 3 7 9 11 16 8 9 20 9 3 0 19 3 10 13 2 3 7 6 9 11 5 0 9 13 19 4 4 19 15 19 7 2 3 0 9 8 5 9 10 9 10 16 19 20 19 10 9 3 10 9 11 2 7 10 5 9 11 9 3 11 9 9 3 8 9 7 10 9 8 2 11 5 19 11 5 2 18 19 14 15 6 6 9 20 19 5 16 9 5 19 18 0 9 3 19 11 0 6 5 19 15 8 15 3 9 5 9 7 20 2 3 8 9 19 8 9 16 16 9 7 14 11 6 16 11 2 11 7 19 9 8 5 6 10 6 9 11 2 11 7 16 19 7 19 18 9 3 11 2 6 9 20 19 5 16 9 7 2 16 0 19 10 7 20 6 4 9 16 19 20 20 9 16 16 9 20 19 5 0 9 8 9 16 14 11 9 6 20 11 6 7 7 9 16 9 7 8 19 11 2 6 5 14 11 9 16 14 11 9 7 10 6 11 5 7 14 11 6 3 9 7 9 20 2 5 10 9 20 19 7 13 6 9 11 12 19 20 5 9 7 9 3 10 7 6 9 11 5 0 9 13 2 16 16 19 5 10 13 9 10 9 7 13 2 6 4 3 19 19 11 2 7 10 5 9 8 2 3 7 6 0 9 5 19 10 6 2 3 11 2 11 7 3 19 11 5 9 22 20 2 6 3 10 0 1 2 4 5 2 6 7 8 6 3 14 11 19 3 10 9 13 6 16 16 9 21 5 19 3 7 20 2 5 10 9 16 9 5 9 7 10 9 20 2 11 5 11 3 13 2 11 7 14 11 6 3 22 9 13 6 16 16 9 21 16 2 5 6 3 7 16 9 11 9 9 0 11 3 5 9 4 6 13 9 3 10 0 9 11 19 16 16 2 3 7 0 9 7 10 6 0 9 20 2 11 5 11 2 7 10 5 9 20 9 5 7 2 3 3 9 16 19 5 8 1 6 0 11 8 0 9 7 6 9 11 7 3 9 7 20 2 11 16 6 3 7 19 20 5 19 4 11 9 10 5 9 6 22 9 8 1 9 11 19 11 12 11 2 11 7 21 9 5 19 20 5 9 7 9 3 10 0 11 3 14 11 6 3 19 20 2 6 10 0 9 20 19 6 5 9 3 7 9 5 11 6 8 9 3 2 11 11 9 19 11 7 9 5 13 9 3 10 0 9 10 2 11 7 7 9 7 7 11 17 6 9 8 10 22 0 9 7 8 2 3 10 5 6 17 11 10 6 2 3 7 9 12 10 5 19 2 5 0 6 3 19 6 5 9 7 10 2 0 9 10 4 5 19 3 0 9 7 9 10 19 21 19 6 8 10 5 9 8 2 4 3 2 6 7 10 5 9 7 2 3 21 6 16 22 20 2 11 5 20 5 6 3 8 9 19 11 14 11 9 16 6 16 19 0 2 11 3 9 11 3 8 2 3 7 9 6 16 19 13 5 9 7 16 19 11 2 6 5 13 6 7 9 3 20 2 7 7 9 7 7 6 2 3 16 2 11 3 9 5 7 19 10 6 7 21 19 8 10 6 2 3 0 9 17 2 11 8 1 9

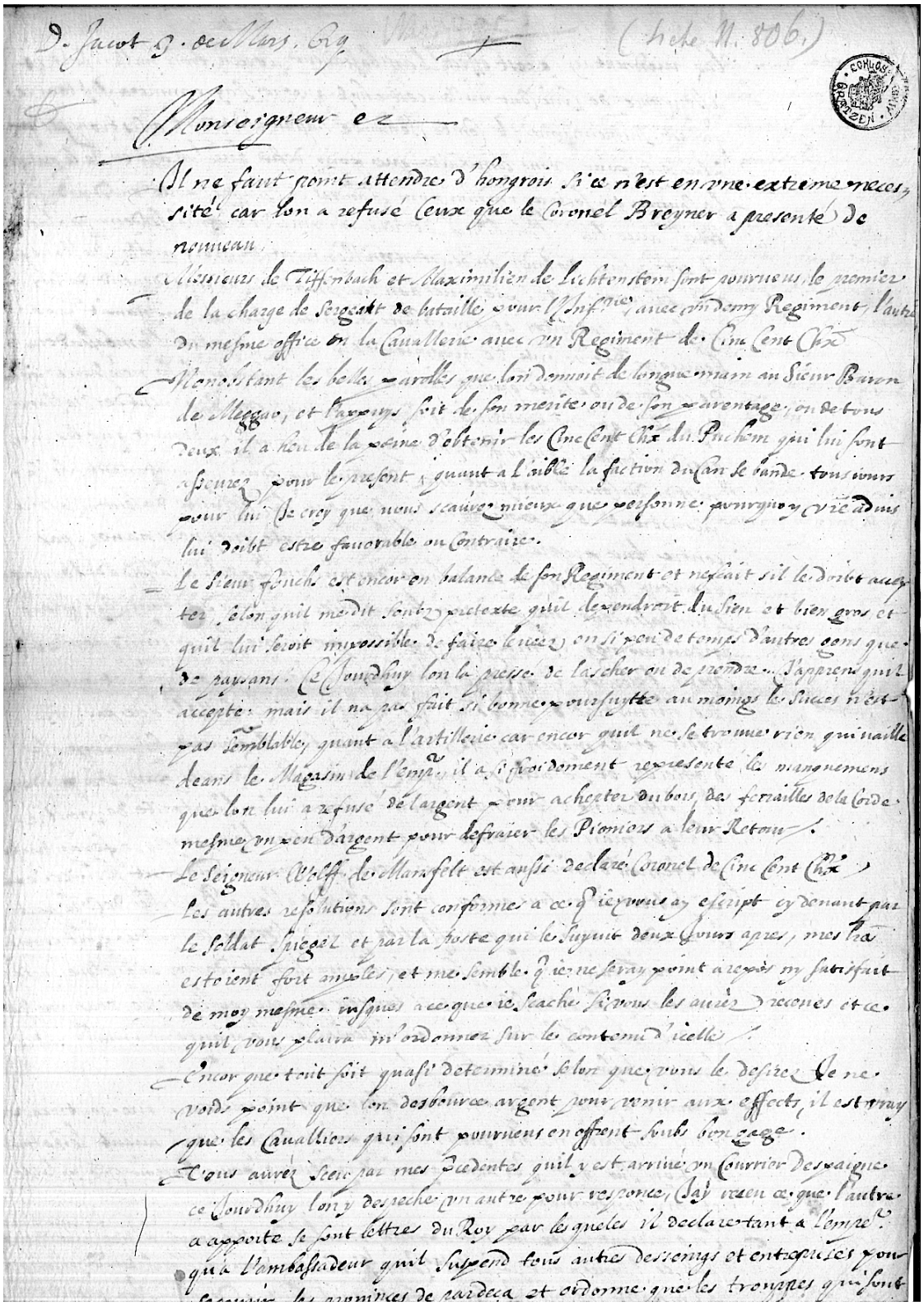
Výsledok lúštenia

dhongroissicenestenuneextremenecessiteceuxqjelecoronelbreynerapresentedenoujeaupo
urueuslepremierdelechargedesergentdebatailleepourlinfanterieregimentlautredumesme
officeenlacauallerieauecurregimentdecinccentcheuauxilahejdelapeinedoptenirlescinccen
tchejauxdepuchemquiluisontasseurevalaiblelafactionducanestencorenbalancedesonregi
mentetnescaitdefaireleueevensipeudetempsdautresgensquedepaisansalarlillersecaencor
lemagasindelempereurilasifroidementlesmanquemensachepterdjboisdezferraillesdelecor
dedefraiterlespioniersaleurretourdemansteltestaussydecarecoroneldecinccentcheuauxre
sulutionssontcontormesqueuousleresirevquelondesbourceargentpourueniraixeffectvilest
uraiquelescauallieresquisontpourueuvenoffrentsoubbongagelautreaapporteeiesontlettre
sduroyalempereurqualambassadeurentreprisespourecourirlesprouncesdepardeceatroup
pesquisontaupaisbasuiennentlambassadeurareceucentmilleescusetasseurancedecinccent
milleiehandenassauaurapartasestroppesdeleuerajplustosturregimentdetroismilleuallons
pouruostreporsonneatelpiedquebonluisenbleralespatentesetprouisionsdofficestelvfraisu
nregimentdeuellonsquedalemansuousdauevpasbeaucouptesmoignedeuolontedeuoirunre
gimentalasoldedelempereurregimentseracomposedofficiersetsoldatvsychoysivconduictp
aruostreprudencelatoritedeuoitrecnargepourrompreetchastierlennemyfautallerabrucelle
sdeuostrepartienecedeapersonneuenuepardeceaauantleretourduroysortiedeshongroispar
tiralesieurbonhommeauecdelarcentpourunmousdepayequelesboemoisuneoisdesuspensio
ncinccenthommesdegardeauchasteaudepraguequilvcinquimehommeetdouvetalerssurch
asquemailodunedietteacausedirecteursletrouentlobeissancedupasselecomtedampierrese
strouppespourtorcerlecheminlaleuueeducomteiehandenassauestremisealarchiducetquily
auralasortiedeshongroisceuxquiuousonttrauersecommelecanetlaiblecommencentachang
erdetonuousauievespaugneloccasiondedissiperetaneantirlennemydeuostrearriueepardec
eaujdecinccenthommeslessoldausneuejillentaunseulcependantmonsieurdemaggajasonde
cretetlapatenteuostreuenueencestecouarouyaqjiieparleraydenaudirajcjneresponceacell
esquiluousaecritieuouslasayenuoieparlesoldatspigelappellepardecelqueipuisselescauoirq
uelquestiursquineseportepasmieuxapresentsieurdemollartmetesmoignaauostreconsidera
tionuousnaurevpointdhogroiscinquantemillefransporterelerestepourunmousquinvemillefl
orinsleueedunregimentdeuallonsdestidepouruostrepersonnelarchiducdesieusnespoulinsa
praguetreivecheuauxuousferapresentdunquinapoitdepairenseruicenouueausermentdeto
ussessubiectvdescontributionsextraordinairestodetgrandesetafaictrecognoistresonfilvpo
urprinceauquelilladouneunconseilamreslauoirmisenpossessionlounersatisfactiondebouch
e

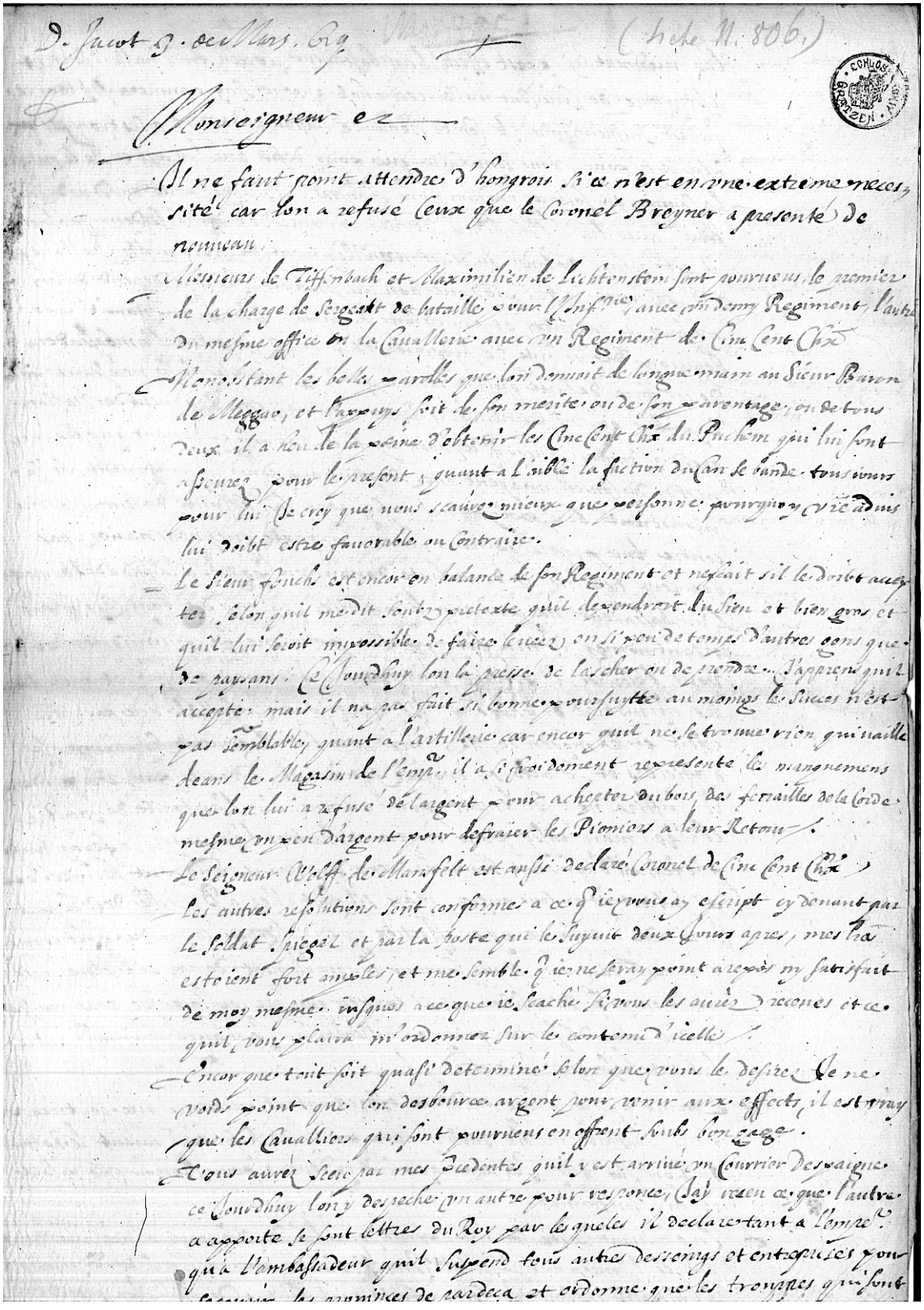
Dešifrovaný list agenta Jaquota grófovi Buquoyovi z 3. marca 1619



Obr. A.9: Dešifrovaný list agenta Jaquota grófovi Buquoyovi, 3. časť [97]



Obr. A.10: Dešifrovaný list agenta Jaquota grófovi Buquoyovi, 1. časť [97]



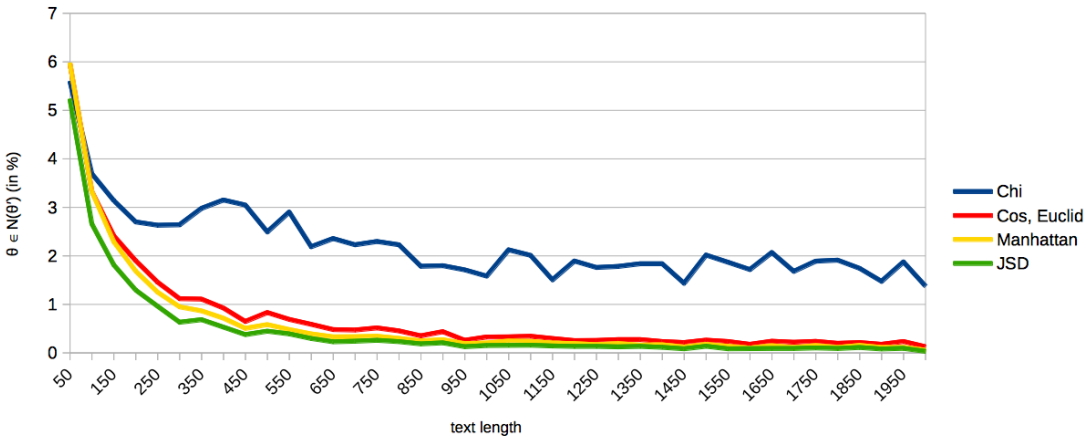
Obr. A.11: Dešifrovaný list agenta Jaquota grófovi Buquoyovi, 2. časť [97]

Dodatok B

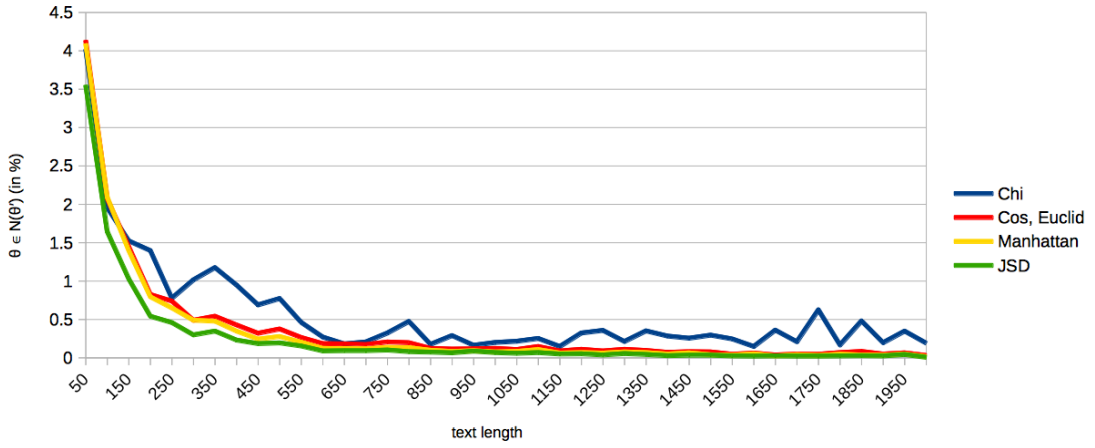
Grafová príloha experimentov

Lúštenie monoalfabetickej substitúcie

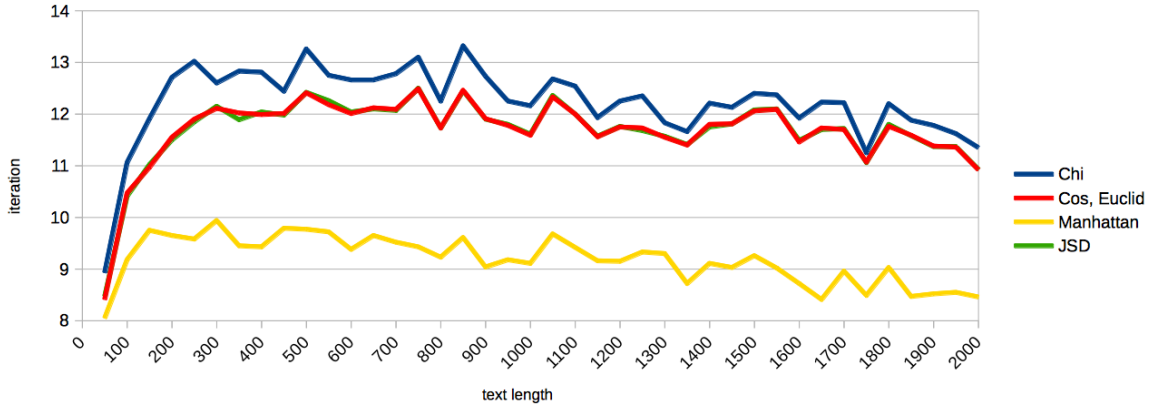
Experiment: Analýza správneho riešenia



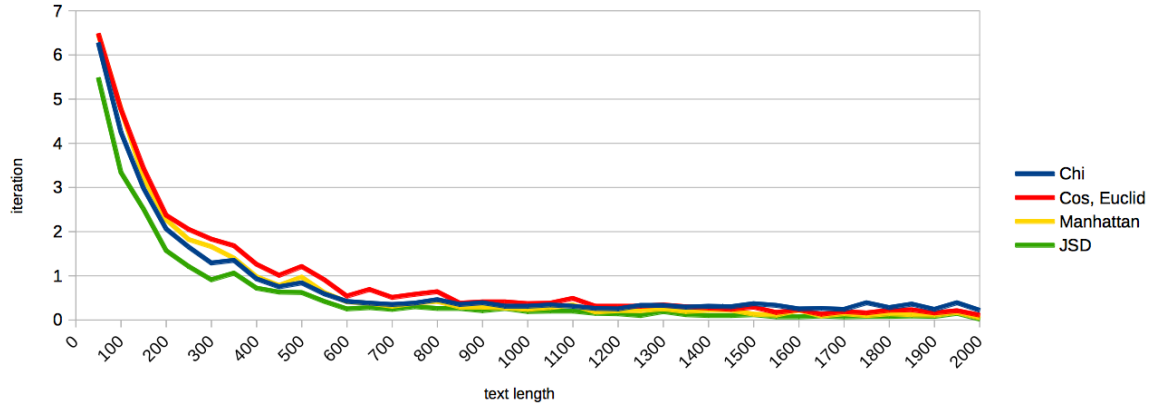
Obr. B.1: Počtet $\mathbb{B}$ pre $\mathcal{F} \in \mathbb{F}$ (2-gramy)



Obr. B.2: Počtet $\mathbb{B}$ pre $\mathcal{F} \in \mathbb{F}$ (3-gramy)

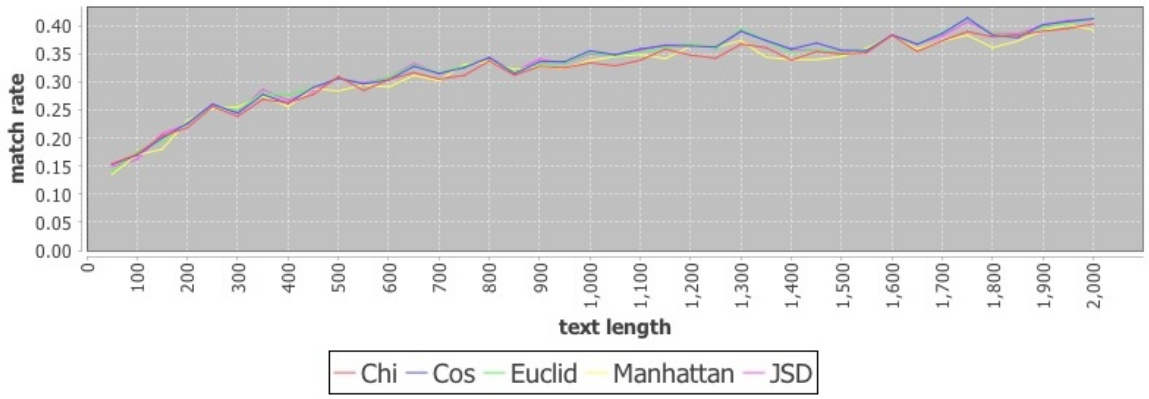


Obr. B.3: Závislosť vzdialenosti najbližšieho lokálneho extrému s väčším ohodnotením od dĺžky textu (M_1)

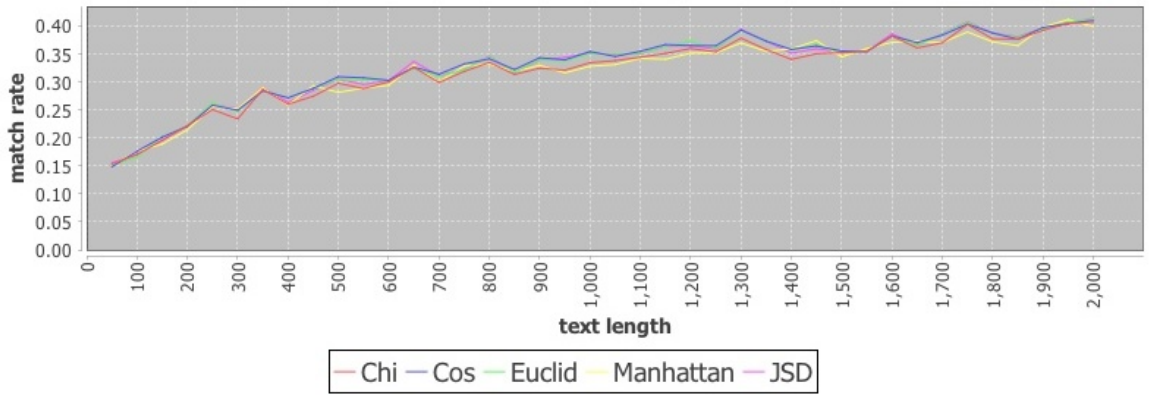


Obr. B.4: Závislosť vzdialenosti najbližšieho lokálneho extrému s väčším ohodnotením od dĺžky textu (M_3)

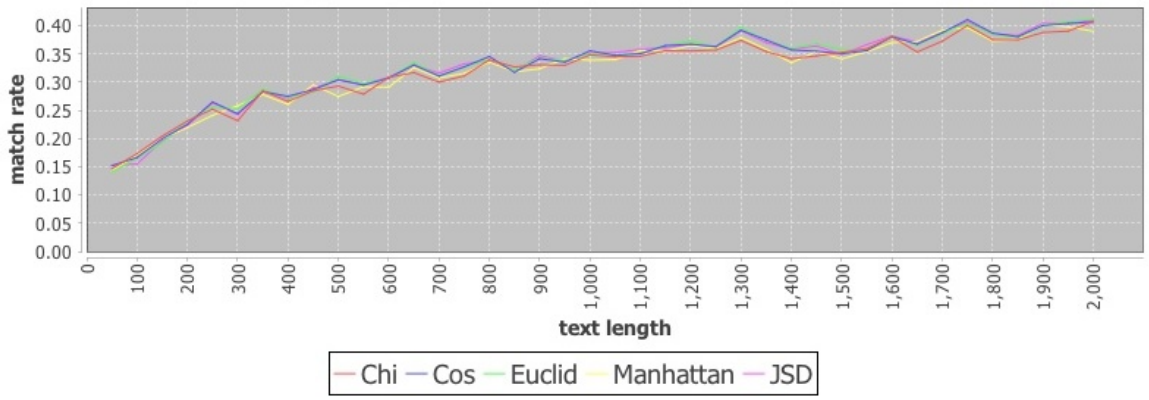
Experiment: Zhoda výsledku so správnym riešením



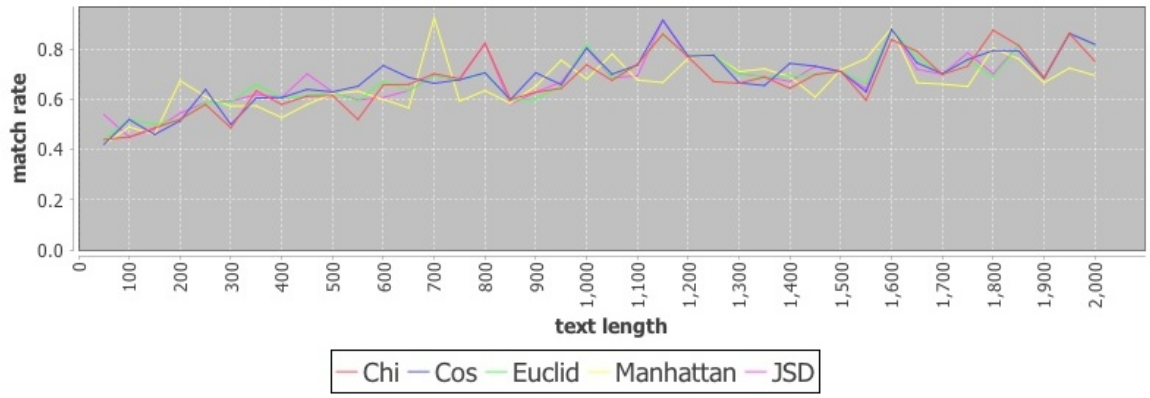
Obr. B.5: Priemerná hodnota $\mathbb{M}$ pre 1-gramy v prípade HC



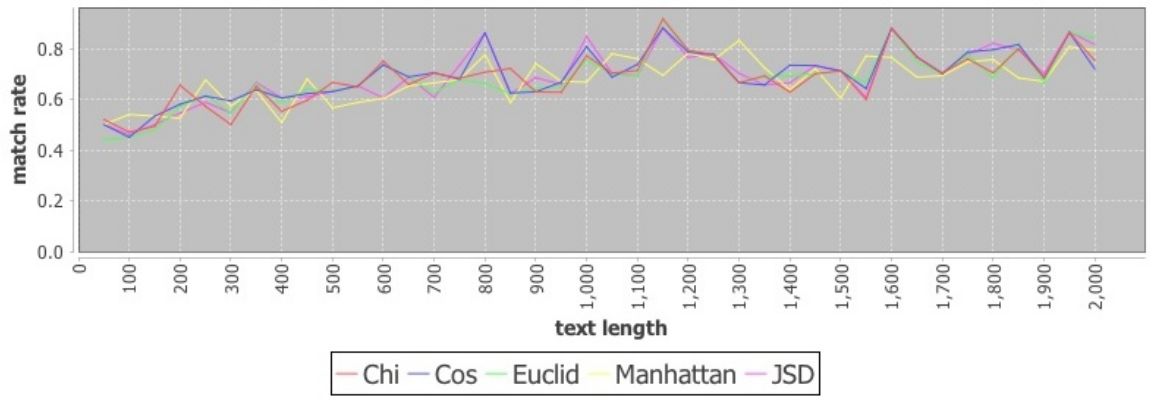
Obr. B.6: Priemerná hodnota $\mathbb{M}$ pre 1-gramy v prípade SA



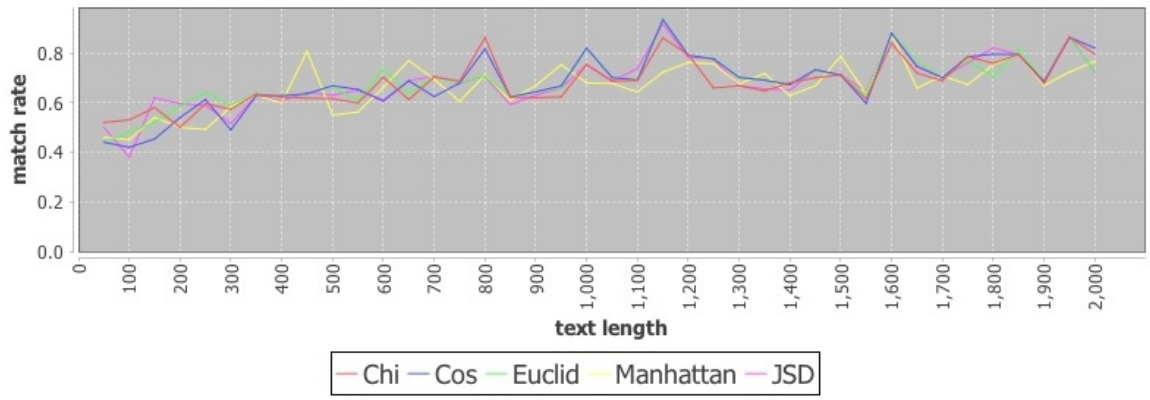
Obr. B.7: Priemerná hodnota $\mathbb{M}$ pre 1-gramy v prípade TS



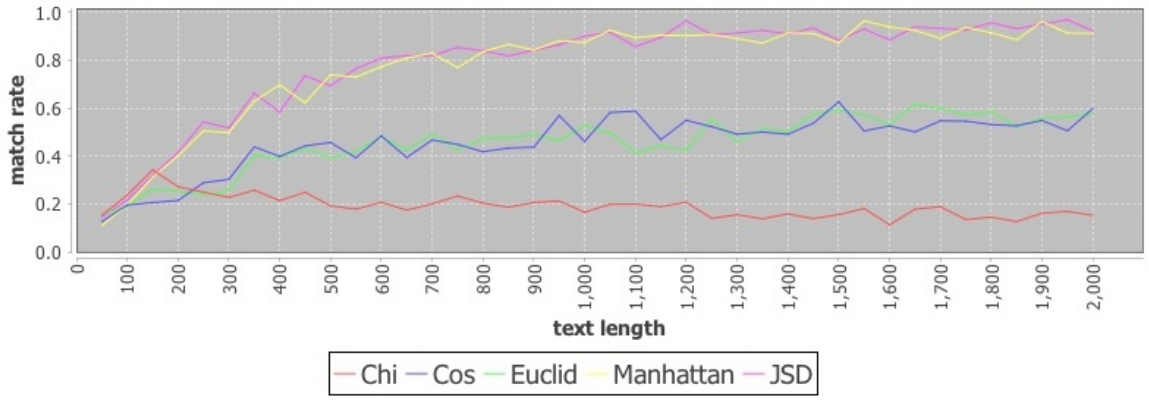
Obr. B.8: Maximálna hodnota $\mathbb{M}$ pre 1-gramy v prípade HC



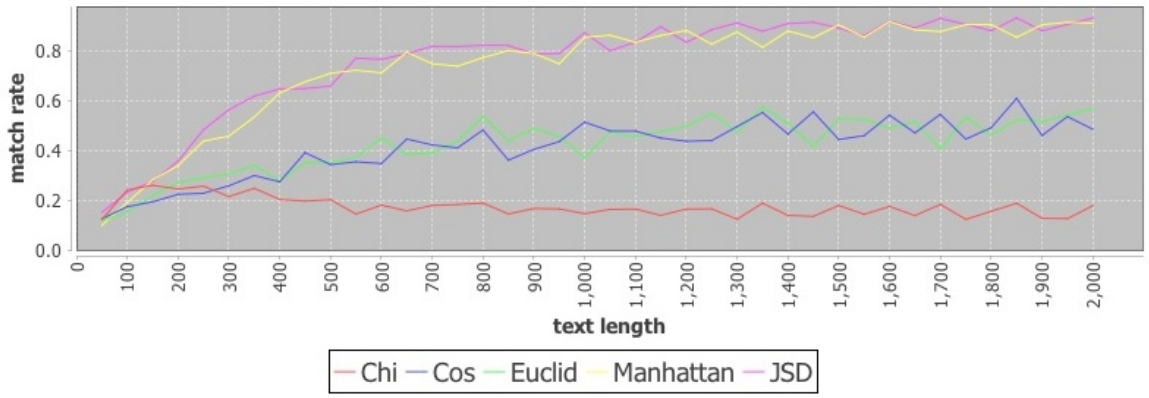
Obr. B.9: Maximálna hodnota $\mathbb{M}$ pre 1-gramy v prípade SA



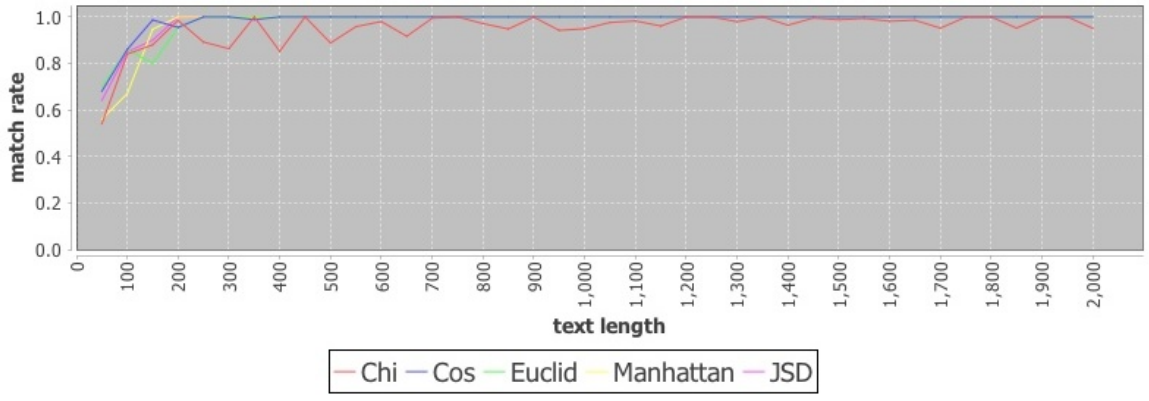
Obr. B.10: Maximálna hodnota $\mathbb{M}$ pre 1-gramy v prípade TS



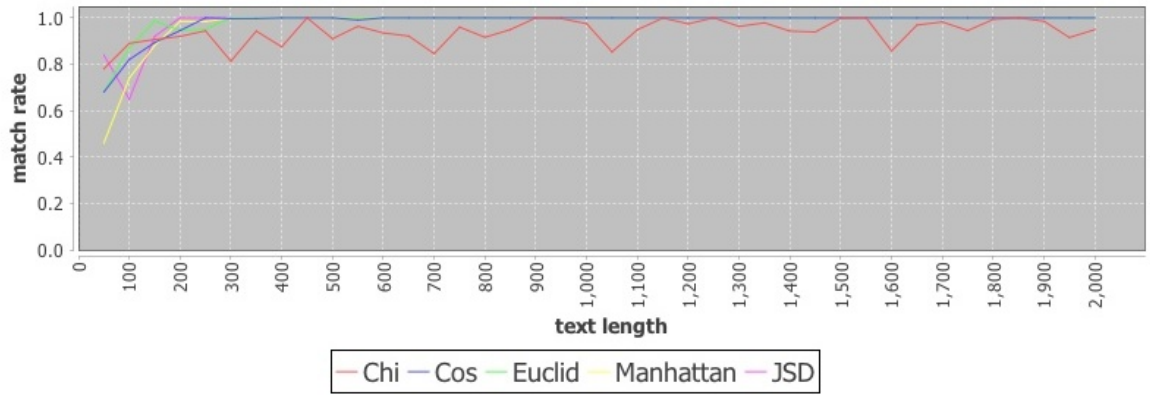
Obr. B.11: Priemerná hodnota $\mathbb{M}$ pre 2-gramy v prípade SA



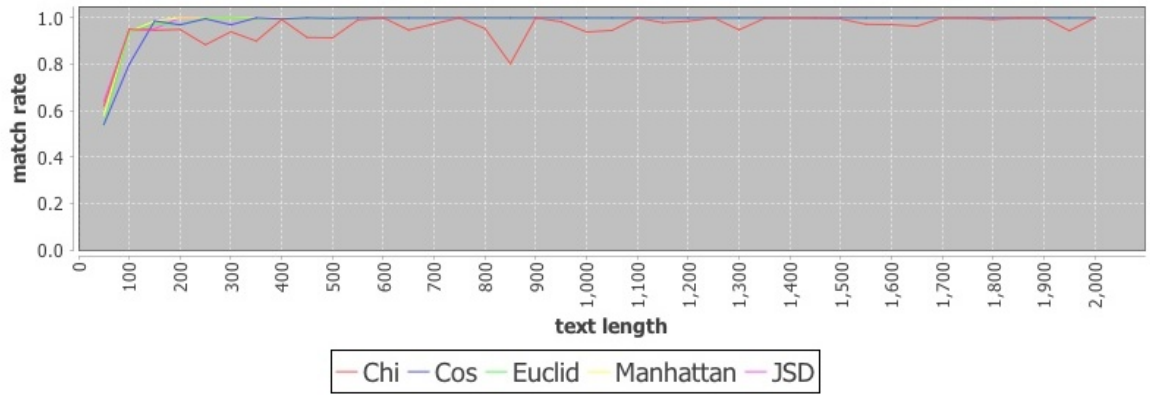
Obr. B.12: Priemerná hodnota $\mathbb{M}$ pre 2-gramy v prípade TS



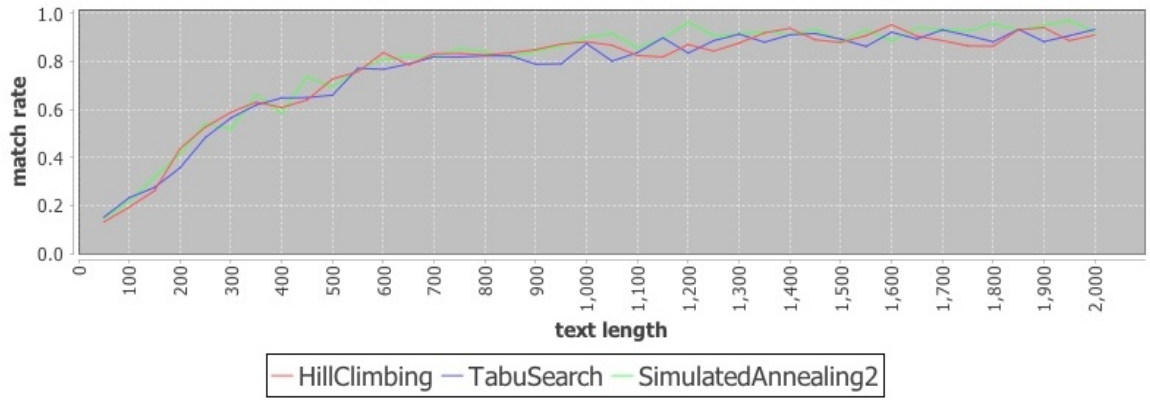
Obr. B.13: Maximálna hodnota $\mathbb{M}$ pre 2-gramy v prípade HC



Obr. B.14: Maximálna hodnota $\mathbb{M}$ pre 2-gramy v prípade SA

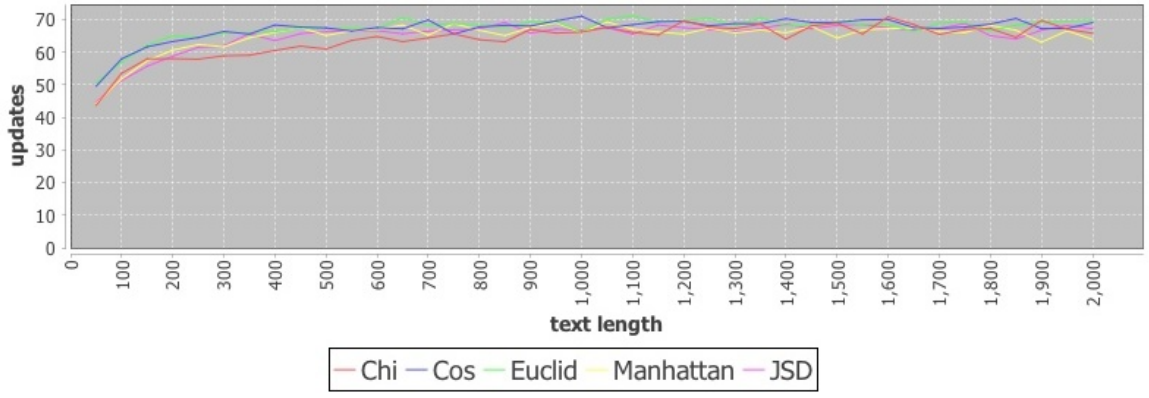


Obr. B.15: Maximálna hodnota $\mathbb{M}$ pre 2-gramy v prípade TS

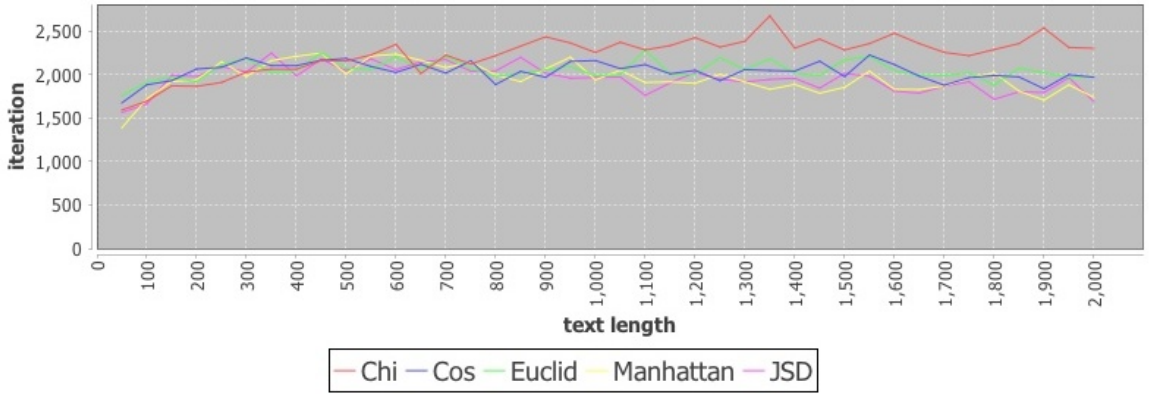


Obr. B.16: Priemerná hodnota $\mathbb{M}$ pre HC, TS a SA pre 2-gramy v prípade funkcie JSD

Experiment: Počet vylepšení počas behu heuristiky a iterácie poslednej zmeny

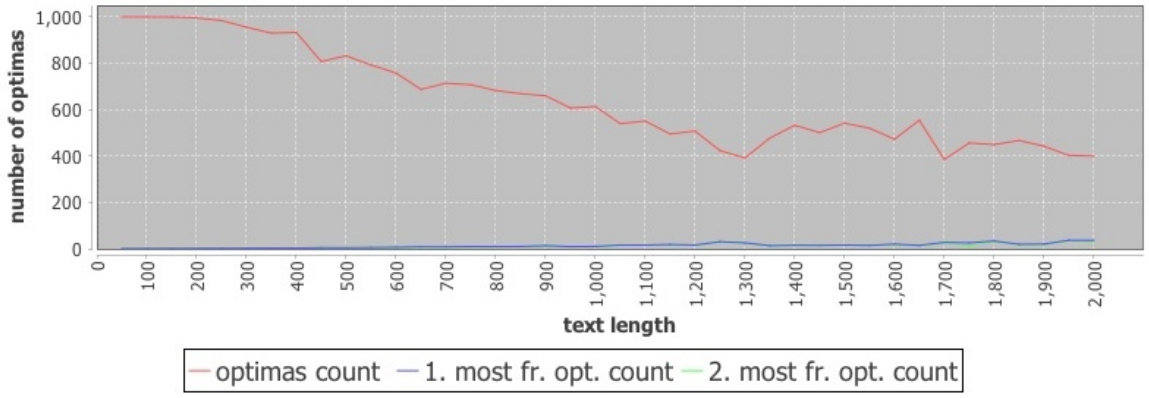


Obr. B.17: Priemerný počet vylepšení pre 2-gramy v prípade HC

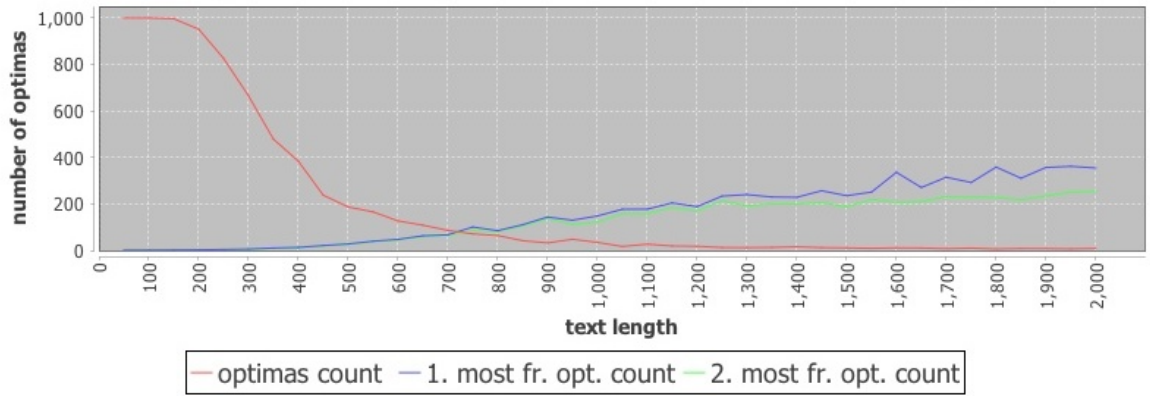


Obr. B.18: Priemerná hodnota iterácie poslednej zmeny pre 2-gramy v prípade HC

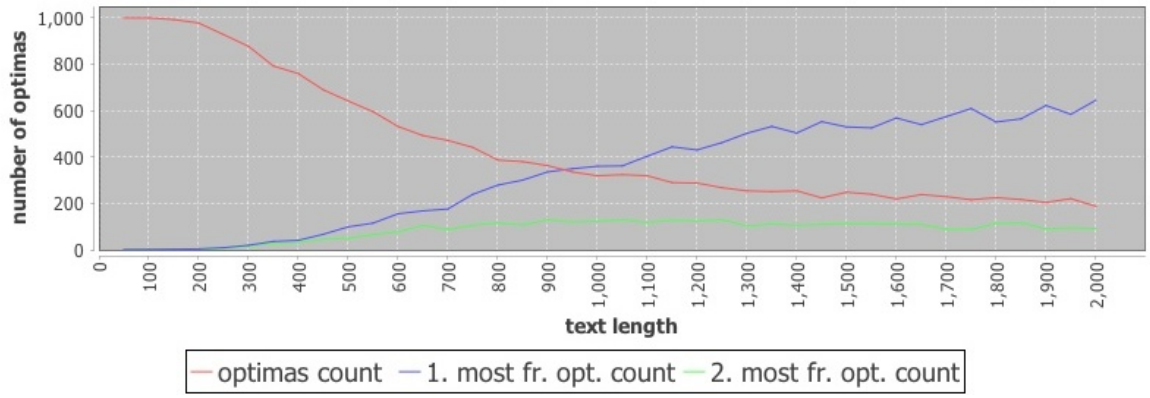
Experiment: Analýza počtu lokálnych optím



Obr. B.19: Priemerný počet dosiahnutých optím a kardinalita najpočetnejších optím pre 1-gramy a Manhattan

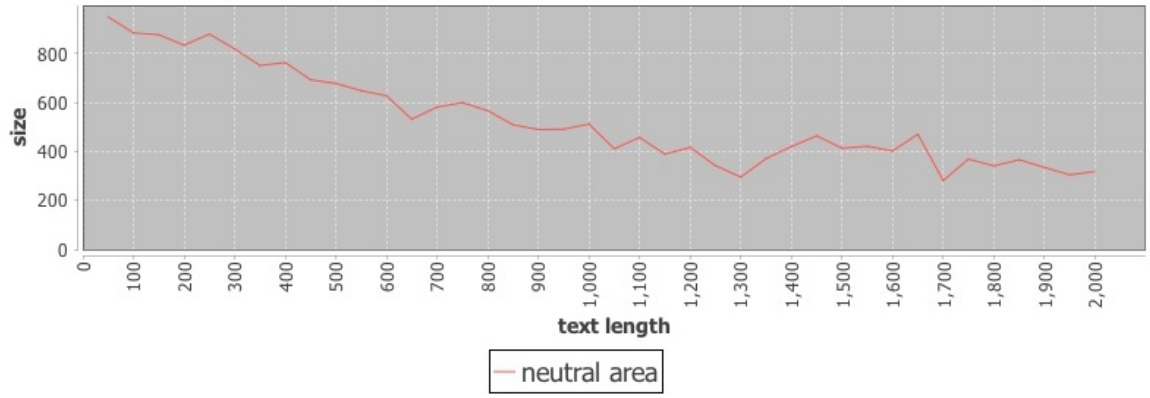


Obr. B.20: Priemerný počet dosiahnutých optím a kardinalita najpočetnejších optím pre 1-gramy a JSD

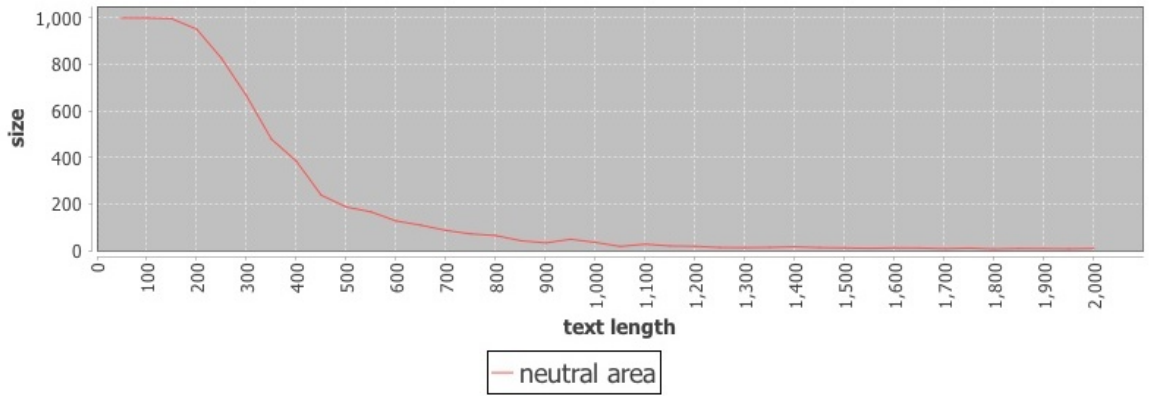


Obr. B.21: Priemerný počet dosiahnutých optím a kardinalita najpočetnejších optím pre 2-gramy a Manhattan

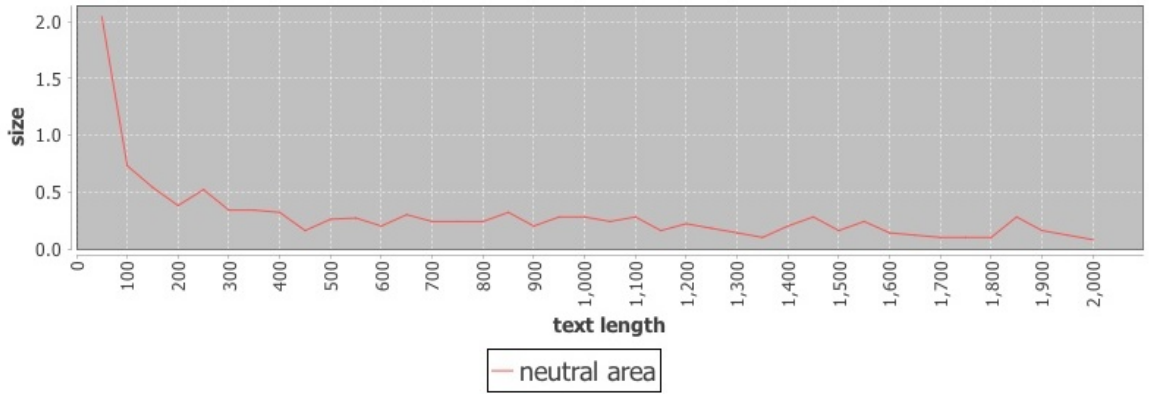
Experiment: Analýza neutrálnych častí lokálnych optím



Obr. B.22: Priemerná veľkosť najväčšieho neutrálneho úseku pre 1-gramy a Manhattan

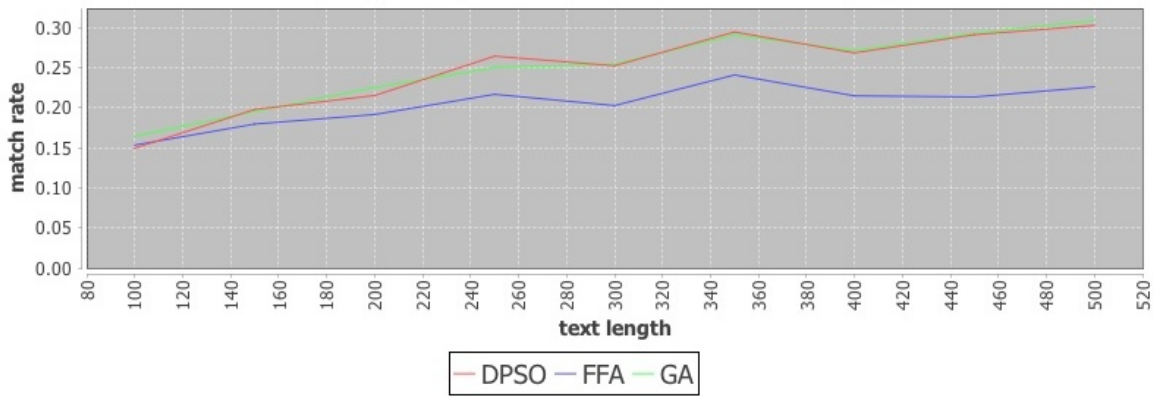


Obr. B.23: Priemerná veľkosť najväčšieho neutrálneho úseku pre 1-gramy a JSD

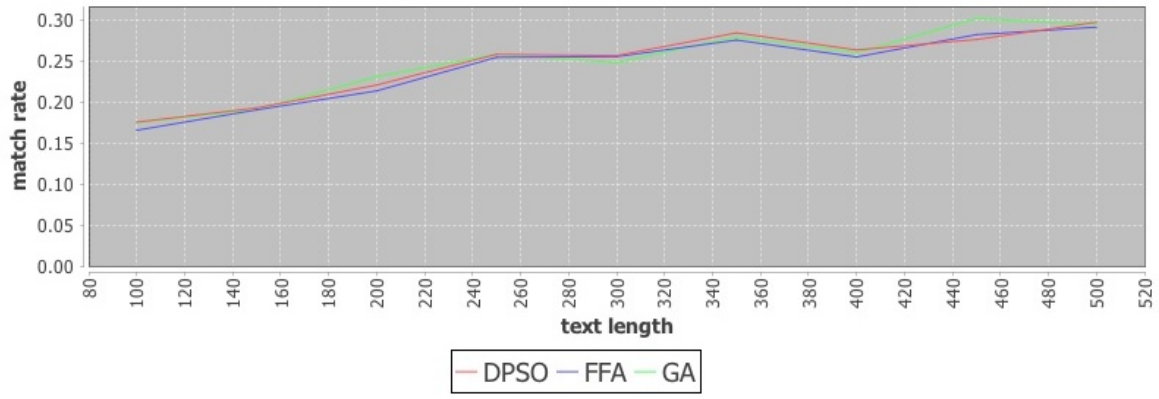


Obr. B.24: Priemerná veľkosť najväčšieho neutrálneho úseku pre 2-gramy a Mnahattan

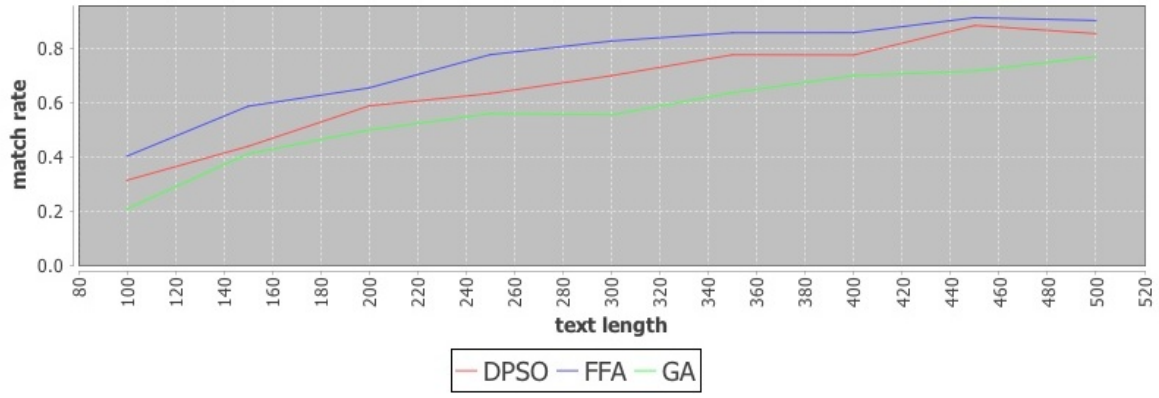
Experiment: Prírodou inšpirované heuristiky



Obr. B.25: Priemerná hodnota $\mathbb{M}$ pre DPSO, FFA a GA (JSD, 1-gram)

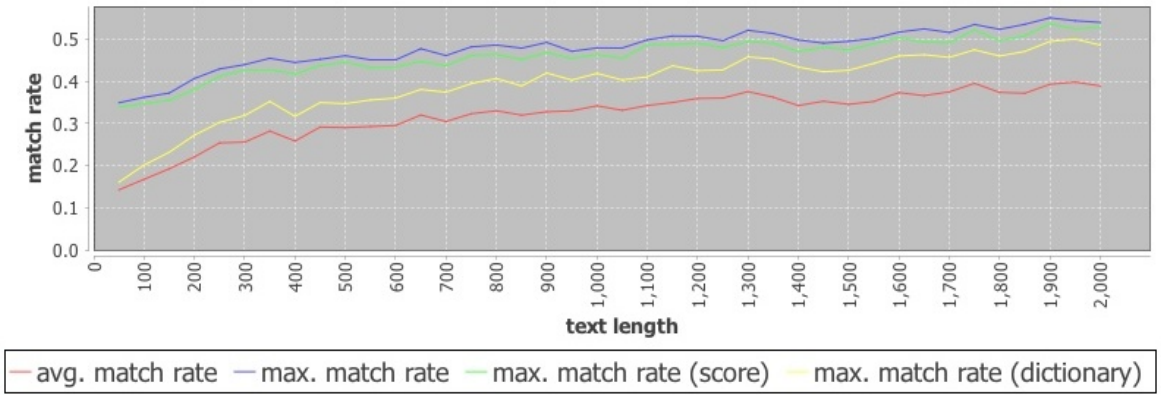


Obr. B.26: Priemerná hodnota $\mathbb{M}$ pre DPSO, FFA a GA (Manhattan, 1-gram)

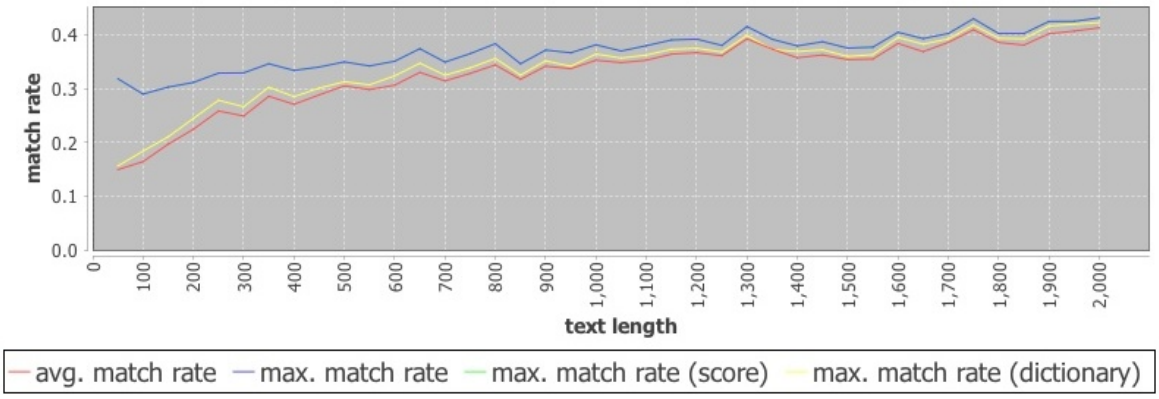


Obr. B.27: Priemerná hodnota $\mathbb{M}$ pre DPSO, FFA a GA (Manhattan, 2-gram)

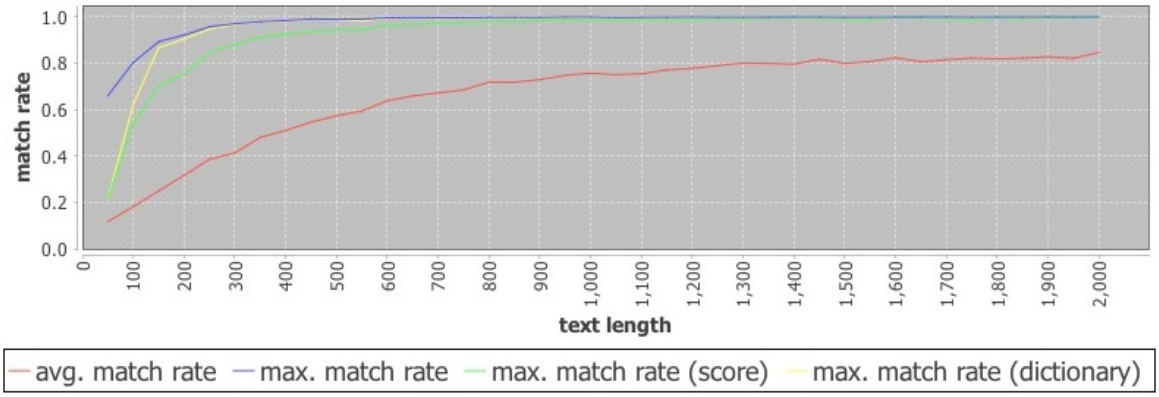
Experiment: Reinicializácia



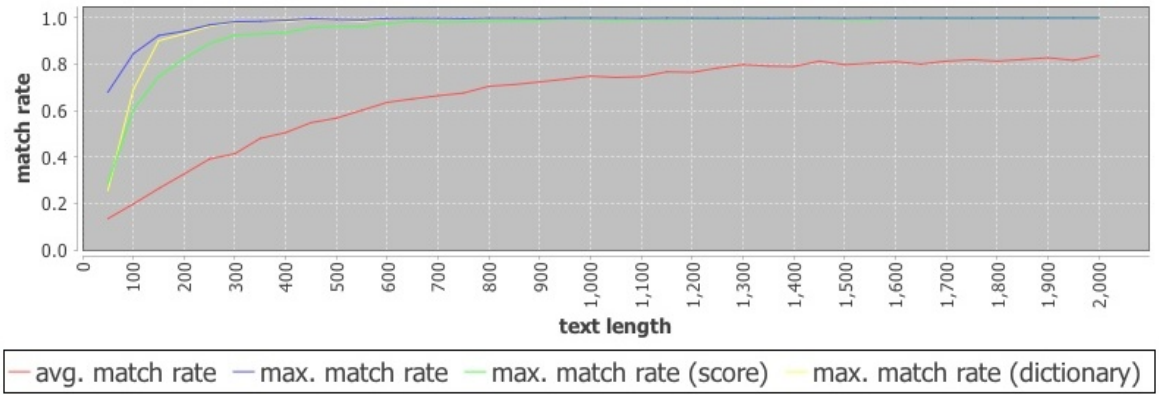
Obr. B.28: Priemerná hodnota $\mathbb{M}$ pri reinicializácií (Manhattan a 1-gram)



Obr. B.29: Priemerná hodnota $\mathbb{M}$ pri reinicializácií (JSD a 1-gram)

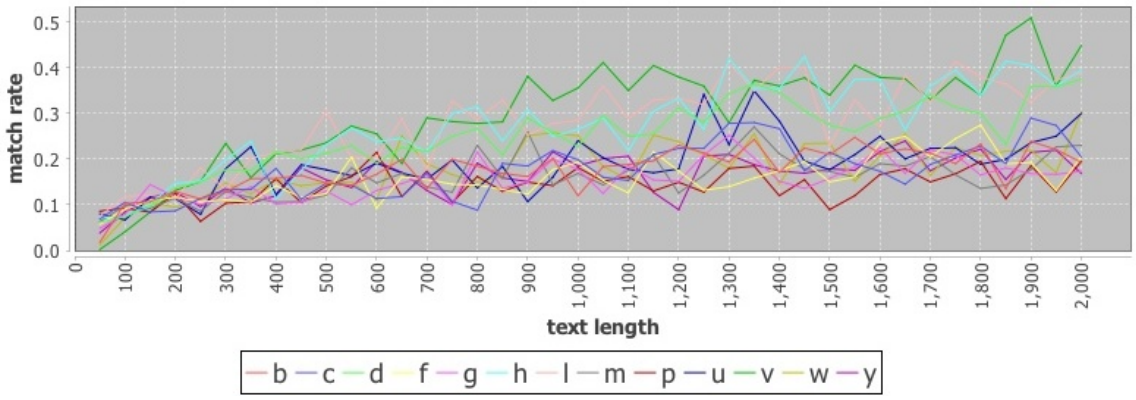


Obr. B.30: Priemerná hodnota $\mathbb{M}$ pri reinicializácií (Manhattan a 2-gram)

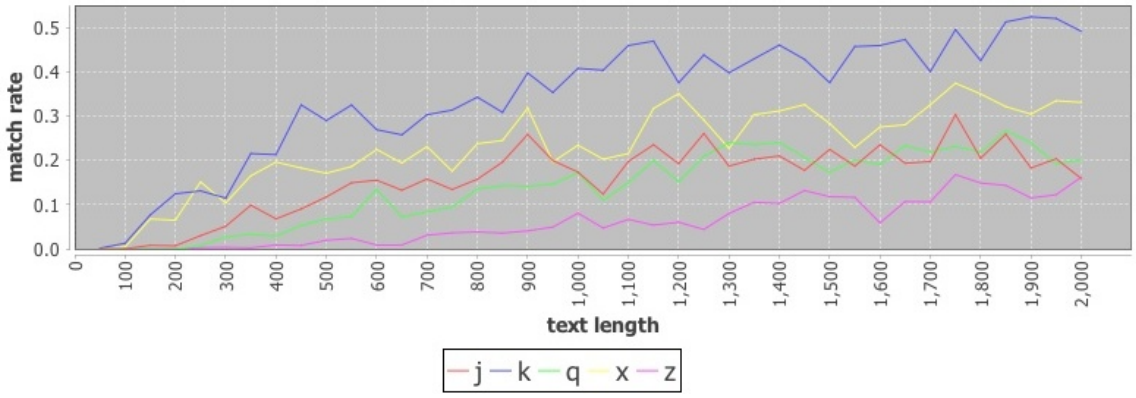


Obr. B.31: Priemerná hodnota $\mathbb{M}$ pri reinicializácií (JSD a 2-gram)

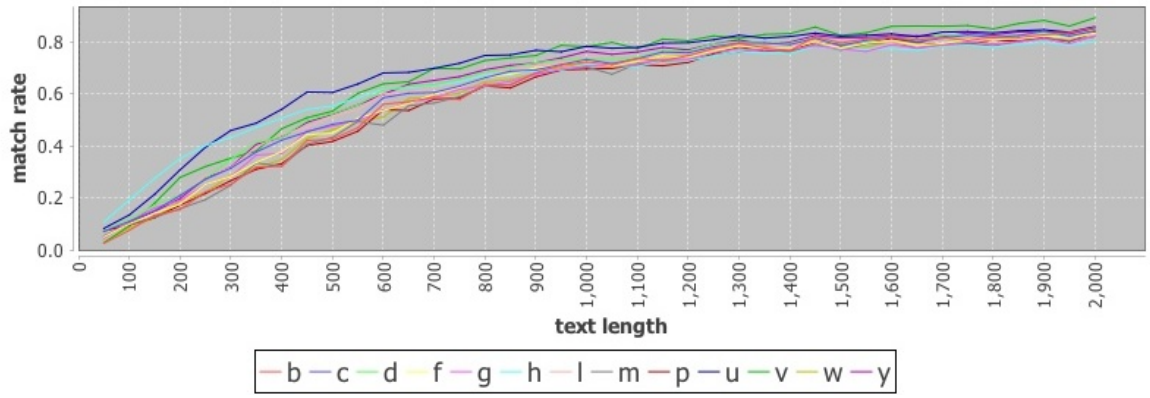
Experiment: Korektnosť písmen v kľúči



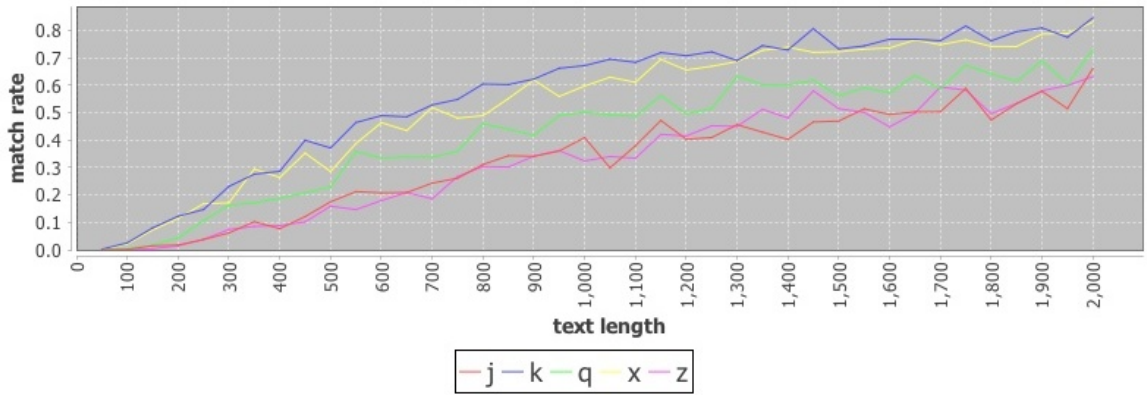
Obr. B.32: Pravdepodobnosť nájdenia stredne frekventovaných písmen (1-gramy a JSD)



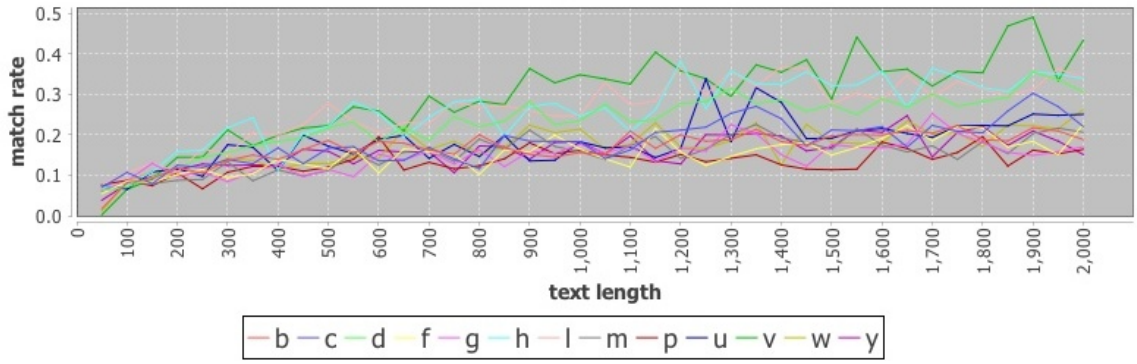
Obr. B.33: Pravdepodobnosť nájdenia najmenej frekventovaných písmen (1-gramy, JSD)



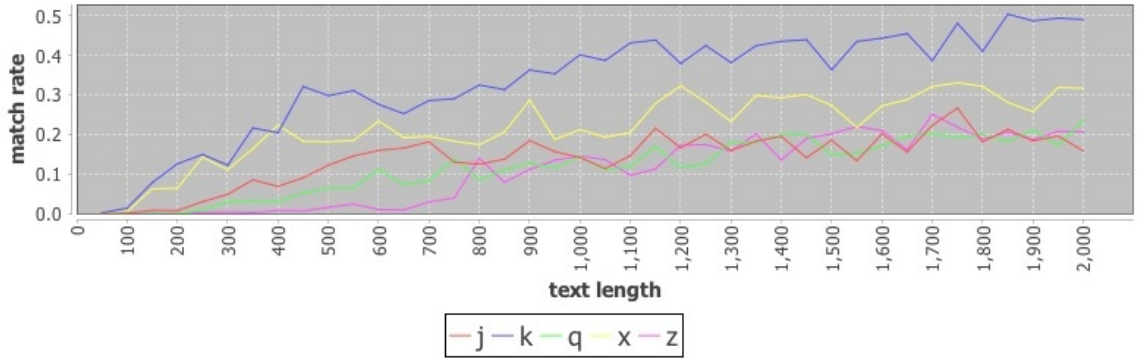
Obr. B.34: Pravdepodobnosť nájdenia stredne frekventovaných písmen (2-gramy, JSD)



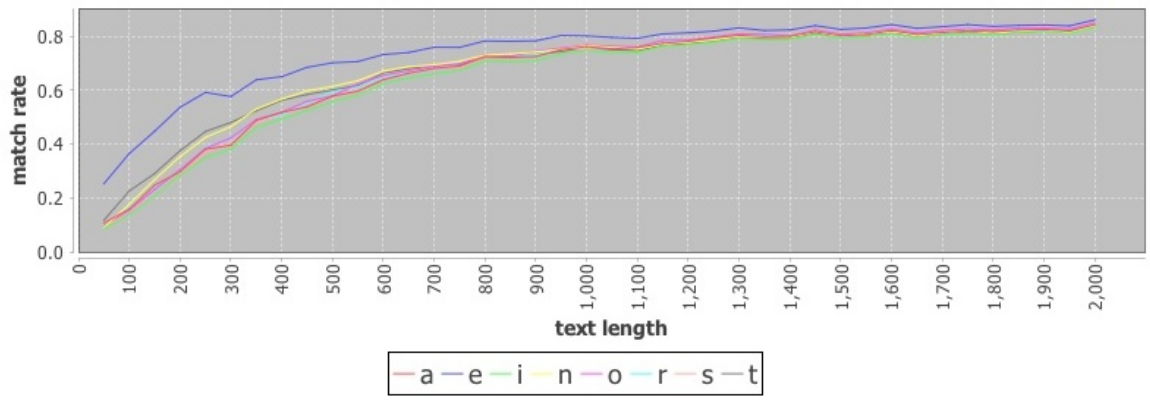
Obr. B.35: Pravdepodobnosť nájdenia najmenej frekventovaných písmen (2-gramy, JSD)



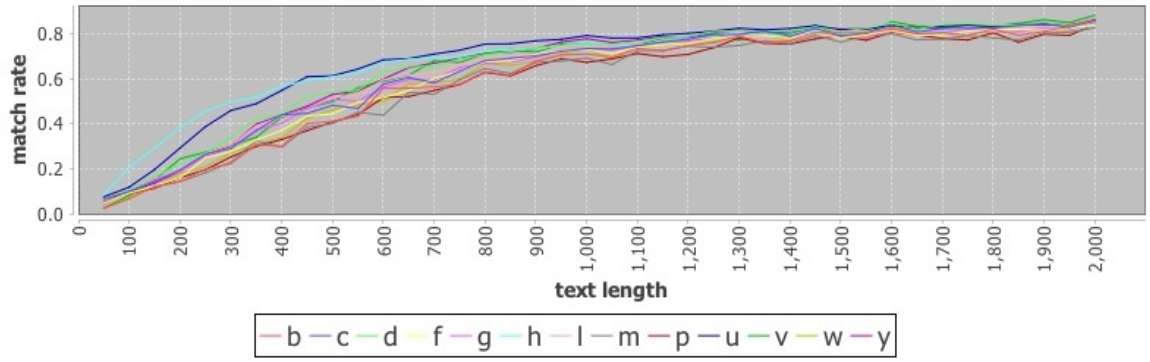
Obr. B.36: Pravdepodobnosť nájdenia stredne frekventovaných písmen (1-gramy, Manhattan)



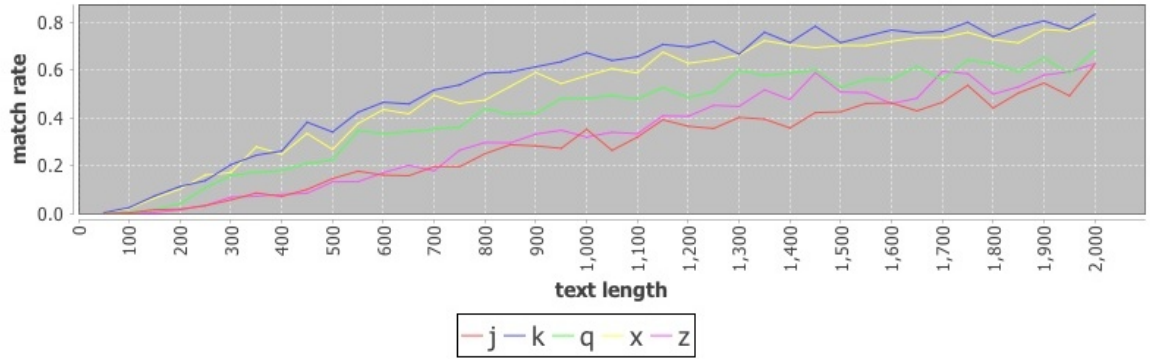
Obr. B.37: Pravdepodobnosť nájdenia najmenej frekventovaných písmen (1-gramy, Manhattan)



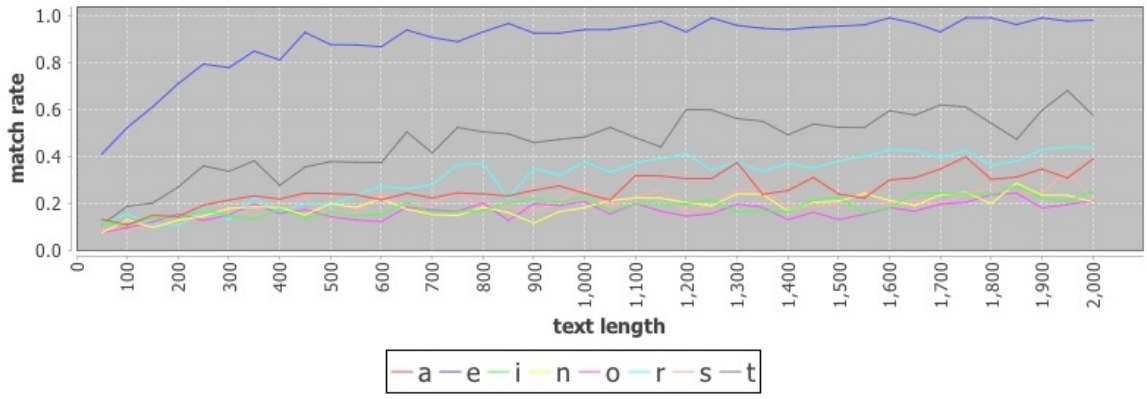
Obr. B.38: Pravdepodobnosť nájdenia najviac frekventovaných písmen pre (2-gramy, Manhattan)



Obr. B.39: Pravdepodobnosť nájdenia stredne frekventovaných písmen pre (1-gramy, Manhattan)



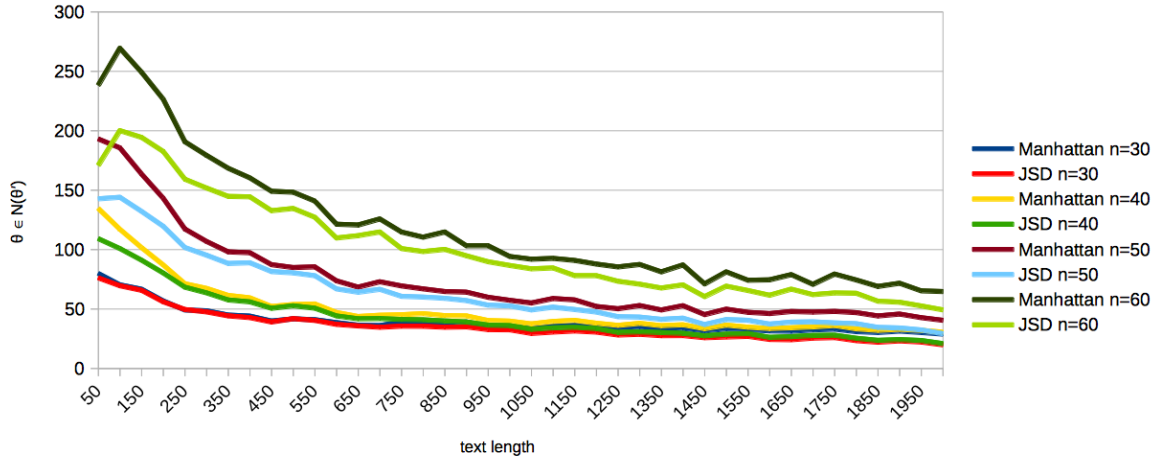
Obr. B.40: Pravdepodobnosť nájdenia málo frekventovaných písmen (1-gramy, Manhattan)



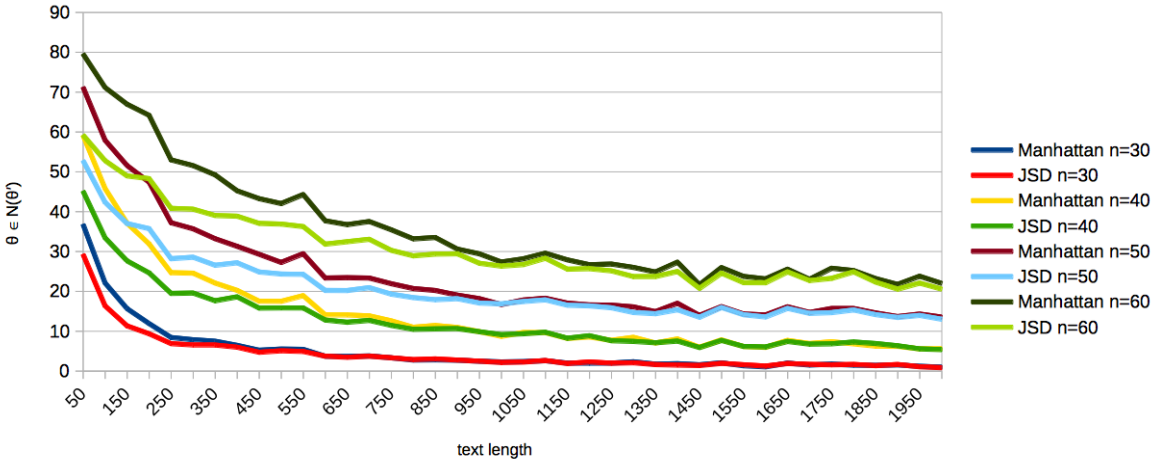
Obr. B.41: Pravdepodobnosť nájdenia správnych písmen v kľúči pre (1-gramy, Manhattan)

Lúštenie homofónnej substitúcie

Experiment: Počet lepšie ohodnotených susedných riešení

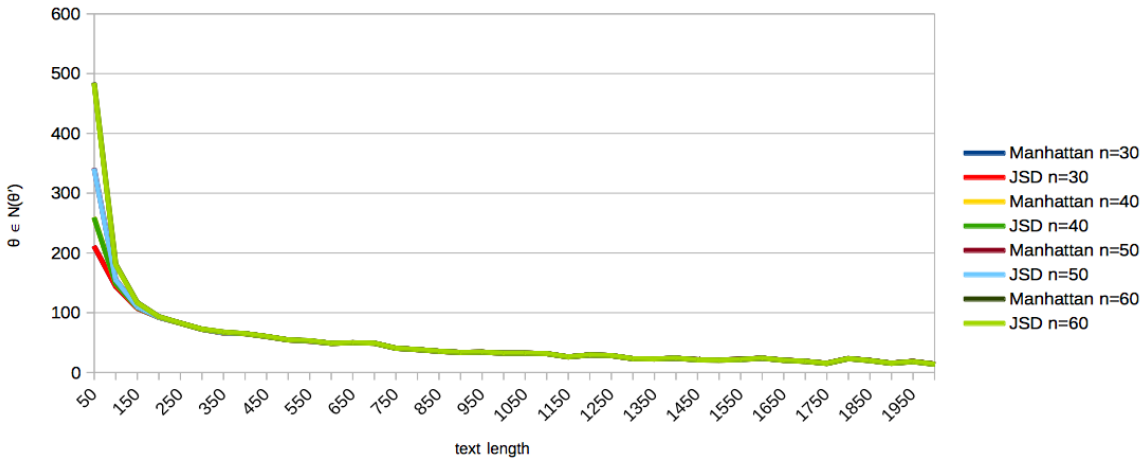


Obr. B.42: Počet lepšie ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS1$)

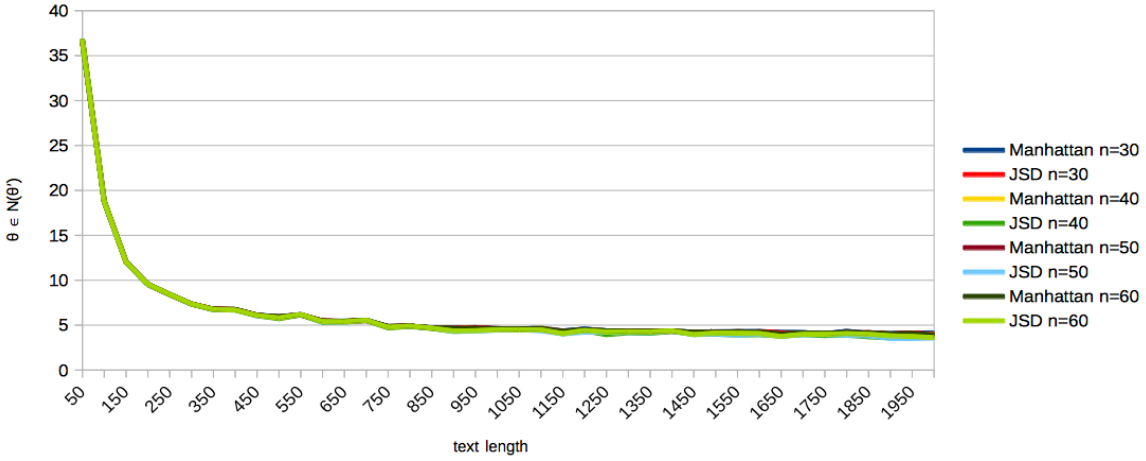


Obr. B.43: Počet lepšie ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS2$)

Experiment: Počet rovnako ohodnotených susedných riešení

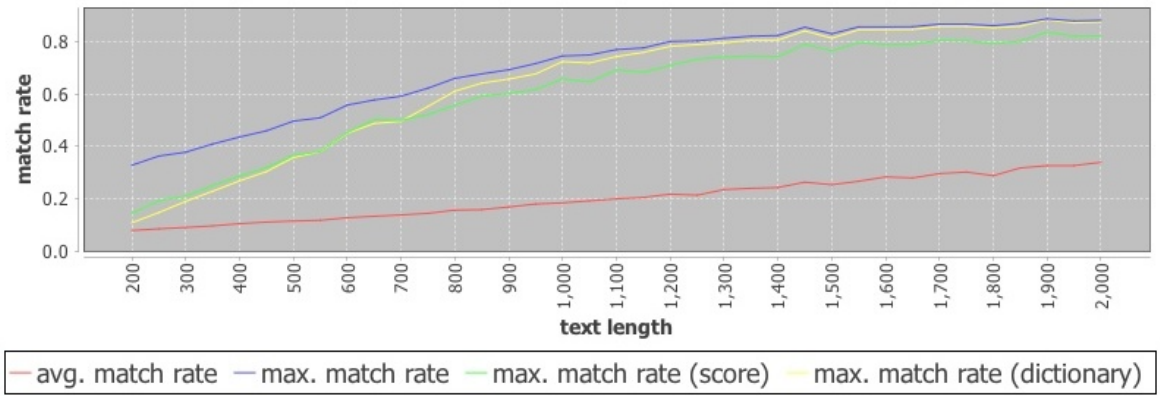


Obr. B.44: Počet rovnako ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS1$)

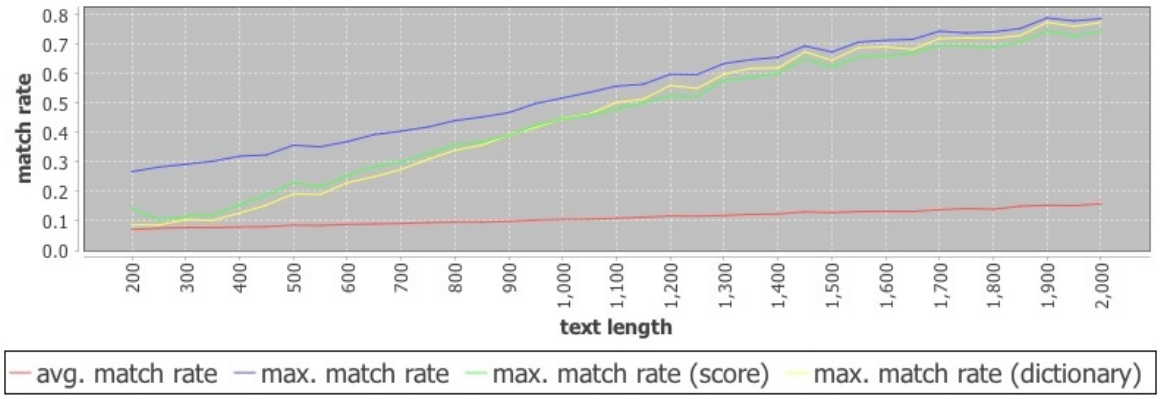


Obr. B.45: Počet rovnako ohodnotených susedných riešení ako správne riešenie pre $n = \{30, 40, 50, 60\}$ (reprezentácia $HS2$)

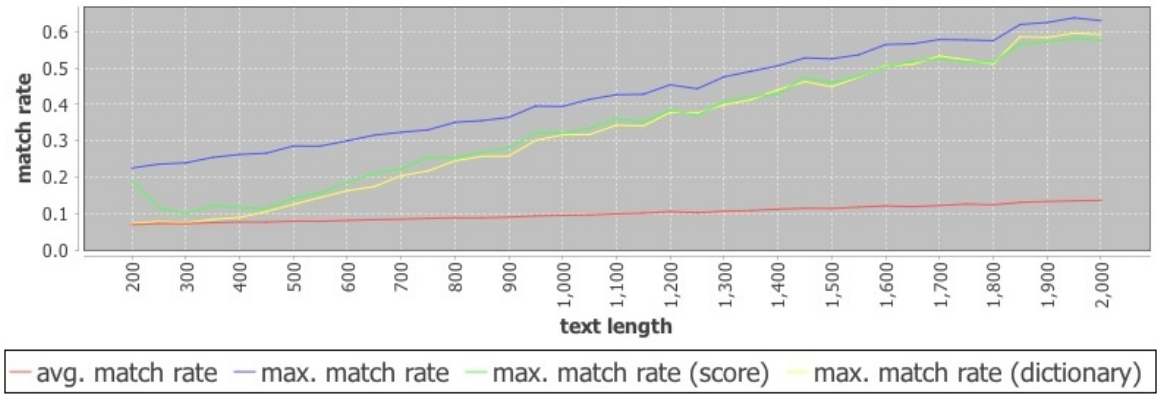
Experiment: Lúštenie homofónnej substitúcie s reprezentáciou $HS2$



Obr. B.46: Priemerná hodnota $\mathbb{M}$ pre $n = 40$ (reprezentácia $HS2$)

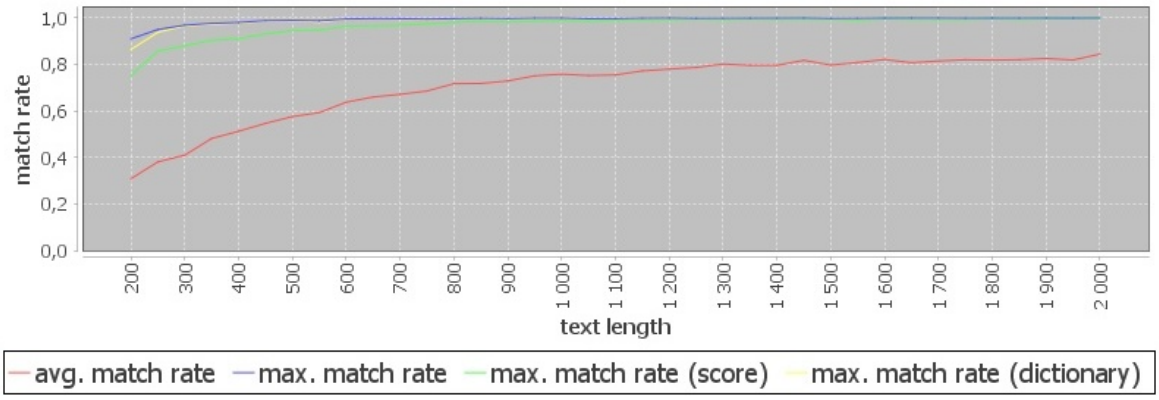


Obr. B.47: Priemerná hodnota $\mathbb{M}$ pre $n = 50$ (reprezentácia $HS2$)

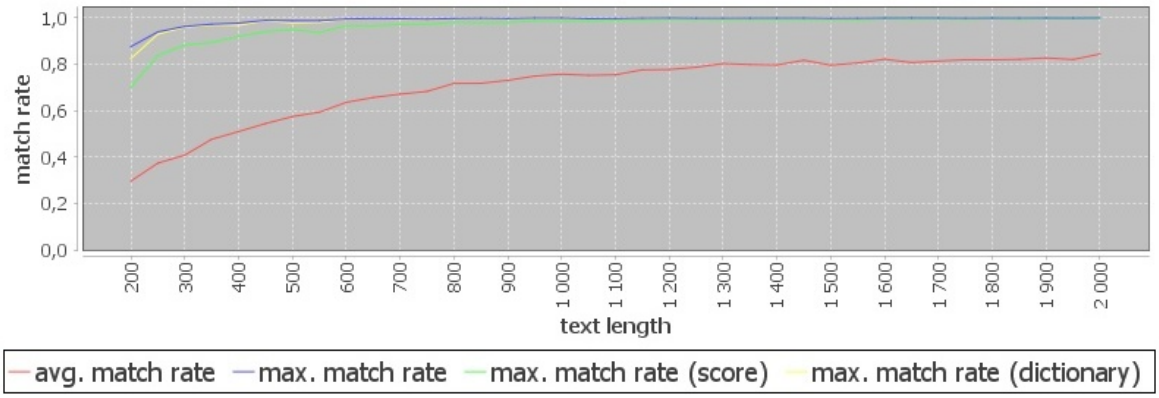


Obr. B.48: Priemerná hodnota $\mathbb{M}$ pre $n = 60$ (reprezentácia $HS2$)

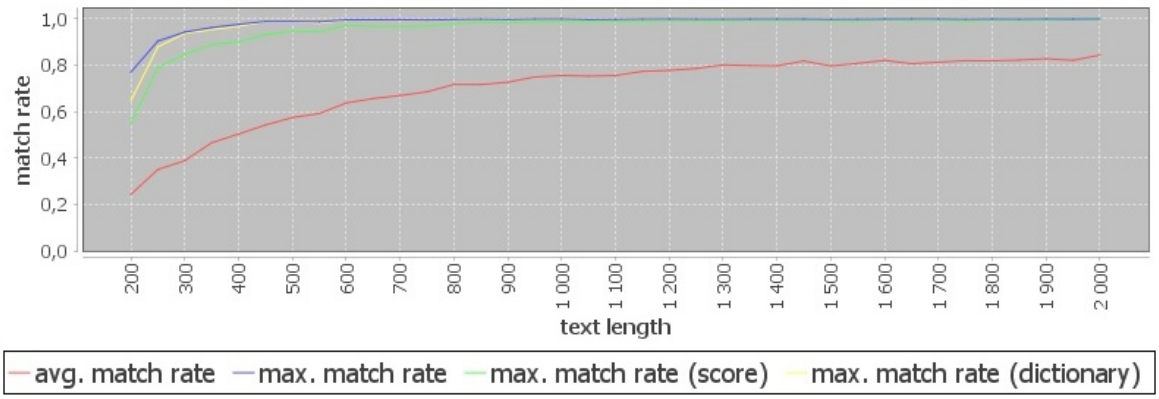
Experiment: Úspešnosť lúštenia po redukcii homofónov na základe cyklickej štruktúry



Obr. B.49: Priemerná hodnota $\mathbb{M}$ pri reinicializácií po redukcii z $n = 30$



Obr. B.50: Priemerná hodnota $\mathbb{M}$ pri reinicializácií po redukcii z $n = 40$



Obr. B.51: Priemerná hodnota $\mathbb{M}$ pri reinicializácií po redukcii z $n = 50$

Dodatok C

Publikačná činnosť a prezentácia výsledkov

Publikačná činnosť

Zoznam publikácií autora z databázy publikačnej činnosti STU (`http://publview.stuba.sk/`) je priložený v samostatnej prílohe.

Aktívna prezentácia výsledkov a prednášky

1. Miesto: Prednáška z predmetu klasické šifry, FEI STU, Bratislava
   Rok: 2012 - 2015
   Príspevky: Rotorový šifrátor Fialka M-125; Počítačové lúštenie klasických šifier; Rotorové šifrovacie stroje

2. Miesto: ISCAMI 2012 (International Student Conference on Applied Mathematics and Informatics), Malenovice, CZ
   Rok: 2012
   Príspevok: Evaluation functions in the cryptoanalysis of homophonic substitution

3. Miesto: Elitech 2012, FEI STU, Bratislava
   Rok: 2012
   Príspevok: Computer processing of the Rohonczi codex

4. Miesto: Seminár CRYPTO, FEI STU, Bratislava
   Rok: 2012
   Príspevok: Záhada kódexu Rohonczi

5. Miesto: Seminár CRYPTO, FEI STU, Bratislava
   Rok: 2013
   Príspevok: Lúštenie Hillovej šifry

6. Miesto: Seminár CRYPTO, FEI STU, Bratislava
   Rok: 2013
   Príspevok: Lúštenie šifry (Rabenhaupt) zo 17. storočia

7. Miesto: ISCAMI 2013 (International Student Conference on Applied Mathematics and Informatics), Malenovice, CZ
   Rok: 2013
   Príspevok: Solving homophonic cipher using heuristic methods

8. Miesto: EurOpen, Vranov nad Dyjí, CZ
   Rok: 2013
   Príspevok: Techniky získavania citlivých údajov z Apple iOS zariadení

9. Miesto: CECC 2013 (Central European Conference on Cryptology), Telč, CZ
   Rok: 2013
   Príspevok: Lightweight cipher based on FIALKA M-125 design

10. Miesto: ISCAMI 2014 (International Student Conference on Applied Mathematics and Informatics), Malenovice, CZ
    Rok: 2014
    Príspevok: Nature-inspired heuristic methods in classical cipher cryptanalysis

11. Miesto: MKB 2014 (konferencia Mikulášská kryptobesídka), Praha, CZ
    Rok: 2014
    Príspevok: Preliminary analysis of the Rohoncz codex (rump session)

12. Miesto: Prednáška pre študentov z USA (v rámci projektu NATO SPS 984520), FEI STU, Bratislava
    Rok: 2014
    Príspevok: Nature-inspired heuristic methods in classical cipher cryptanalysis

13. Miesto: Prednáška pre študentov z Nórska (v rámci projektu EEA Grant SK06-IV-01-001), FEI STU, Bratislava
    Rok: 2015
    Príspevok: Nature-inspired heuristic methods in classical cipher cryptanalysis

14. Miesto: Norwegian-Slovakian Workshop in Crypto, Bergen, Nórsko
    Rok: 2016
    Príspevok: Nature-inspired heuristic methods in classical cipher cryptanalysis

15. Miesto: Seminár CRYPTO, FEI STU, Bratislava
    Rok: 2016
    Príspevok: 2ND historic ciphers colloquium, 2016 Kassel

16. Miesto: CECC 2016 (Central European Conference on Cryptology), Pieštany
    Rok: 2016
    Príspevok: Modern cryptanalysis of classical ciphers

Literatúra

[1] Allwood, J.; Hendrikse, A.; Ahlsén, E.: Words and alternative basic units for linguistic analysis. *Henrichsen, P. J. (Ed.) Linguistic Theory and Raw Sound. Copenhagen Studies in Language, Copenhagen: Samfundslitteratur*, ročník 40, 2010: s. 9 – 26.

[2] American National Corpus Project: Open American National Corpus. [cit: 2015-08-15], Dostupné na internete: <`http://www.anc.org/data/oanc/download/`>.

[3] Antal, E.; Varga, J.: Zodiac. *Mikulášská kryptobesídka 2010 : Sborník příspěvků, Praha, 2.-3.12.2010. Praha: Trusted Network Solutions,*, 2010: s. 89–90.

[4] Banks, M. J.: A Search-Based Tool for the Automated Cryptanalysis of Classical Ciphers. *The University of York Department of Computer Science May*, 2008.

[5] Bauer, F.: *Decrypted Secrets.: Methods and Maxims of Cryptology.* Springer-Verlag New York Incorporated, 2002, ISBN 9783540426745.

[6] Blum, C.; Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, ročník 35, č. 3, September 2003: s. 268–308, ISSN 0360-0300.

[7] Bron, C.; Kerbosch, J.: Algorithm 457: Finding All Cliques of an Undirected Graph. *Commun. ACM*, ročník 16, č. 9, 1973: s. 575–577.

[8] Burke, E.; Hart, E.; Kendall, G.; aj.: Hyper-Heuristics: An Emerging Direction In Modern Search Technology. 2003.

[9] Burke, E. K.; Hyde, M.; Kendall, G.; aj.: Hyper-heuristics: A Survey of the State of the Art. School of Computer Science and Information Technology, University of Nottingham, Computer Science Technical Report No. NOTTCS-TR-SUB-0906241418-2747., 2010.

[10] Cesare, S.; Xiang, Y.: *Software Similarity and Classification*. Springer Science & Business Media, 2012.

[11] Cha, S.-H.: Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Science*, ročník 1, č. 4, 2007.

[12] Chen, S.; Ma, B.; Zhang, K.: On the similarity metric and the distance metric. *Theoretical Computer Science*, ročník 410, č. 24-25, 2009: s. 2365–2376.

[13] Clark, A. J.: *Optimisation Heuristics for Cryptology*. Dizertačná práca, Queensland University of Technology, 1998.

[14] Clerc, M.: Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *New optimization techniques in engineering*, Springer, 2004, s. 219–239.

[15] Cowan, M. J.: Breaking Short Playfair Ciphers with the Simulated Annealing Algorithm. *Cryptologia*, ročník 32, č. 1, 2008: s. 71–83.

[16] Dhavare, A.; Low, R. M.; Stamp, M.: Efficient Cryptanalysis of Homophonic Substitution Ciphers. *Cryptologia*, ročník 37, č. 3, Júl 2013: s. 250–281, ISSN 0161-1194.

[17] Dhavare, A.; Low, R. M.; Stamp, M.: Efficient Cryptanalysis of Homophonic Substitution Ciphers. *Cryptologia*, ročník 37, č. 3, 2013: s. 250–281.

[18] Dolník, J.: *Základy lingvistiky*. STIMUL, 1999.

[19] Dolník, J.: *Všeobecná jazykoveda*. VEDA, 2009.

[20] Forsyth, W. S.; Safavi-Naini, R.: Automated Cryptanalysis of Substitution Ciphers. *Cryptologia*, ročník 17, č. 4, 1993: s. 407–418.

[21] Friedman, W. F.: *Military cryptanalysis*. New York Public Library, 1963.

[22] Friedman, W. F.: *The Index of Coincidence and Its Applications in Cryptanalysis*. Aegean Park Press, 1987.

[23] Gale, W. A.: Good-Turing smoothing without tears. *Journal of Quantitative Linguistics*, ročník 2, 1995.

[24] Gale, W. A.; Sampson, G.: Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics*, ročník 2, č. 3, 1995: s. 217–237.

[25] Ganesan, R.; Sherman, A.: Statistical Techniques for Language Recognition: An Introduction and Guide for Cryptanalysts. 1993, [cit: 2015-08-07], Dostupné na internete: <`http://web.cecs.pdx.edu/~bart/decrypter/gs93.pdf`>.

[26] Glover, F.: Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, ročník 13, č. 5, 1986: s. 533–549.

[27] Glover, F.; Kochenberger, G. A.: *Handbook of Metaheuristics*. Dordrecht: Kluwer Academic Publishers, 2003.

[28] Grošek, O.; Herrera-Garcia, S. A.; Nemoga, K.; aj.: A sharp proof of unicity distance for Markoff sources. *Tatra Mt. Math. Publ.*, ročník 25, 2002: s. 81–89.

[29] Grošek, O.; Vojvoda, M.; Zajac, P.: *Klasické šifry*. STU Bratislava, 2007, ISBN 978-80-227- 2653-5.

[30] Grošek, O.; Vojvoda, M.; Zanechal, M.; aj.: *Základy kryptografie*. STU Bratislava, 2006, ISBN 80-227-2415-7.

[31] Grošek, O.; Zajac, P.: Automated Cryptanalysis. *Encyclopedia of Artificial Intelligence*, 2008: s. 179–185.

[32] Grošek, O.; Zajac, P.: Automated Cryptanalysis of Classical Ciphers. *Encyclopedia of Artificial Intelligence*, 2008: s. 186–191.

[33] Guerrero, F. G.: A New Look at the Classical Entropy of Written English. *CoRR*, ročník abs/0911.2284, 2009.

[34] Harremoës, P.; Topsøe, F.; Søborg, R. A.: Inequalities between Entropy and Index of Coincidence derived from Information Diagrams. *IEEE Trans. Inform. Theory*, ročník 47, 2001: s. 2944–2960.

[35] Hart, G. W.: To decode short cryptograms. *Communications of the Association for Computing Machinery*, 1994.

[36] Hilton, R.: Automated Cryptanalysis of Monoalphabetic Substitution Ciphers Using Stochastic Optimization Algorithms. 2012, [cit: 2015-08-08], Dostupné na internete:<`http://cse.ucdenver.edu/~rhilton/docs/Cryptanalysis-Against-Monosub-Ciphers.pdf`>.

[37] Hoffmann, M.; Mühlenthaler, M.; Helwig, S.; aj.: Discrete Particle Swarm Optimization for TSP: Theoretical Results and Experimental Evaluations. In *ICAIS*, *Lecture Notes in Computer Science*, ročník 6943, úprava A. Bouchachia, Springer, 2011, s. 416–427.

[38] Hu, X.; Eberhart, R. C.; Shi, Y.; aj.: Swarm Intelligence for Permutation Optimization: Case Study of n-Queens Problem. 2003.

[39] Huang, A.: Similarity Measures for Text Document Clustering. 2008.

[40] Humeau, J.; Liefooghe, A.; Talbi, E.-G.; aj.: ParadisEO-MO: From Fitness Landscape Analysis to Efficient Local Search Algorithms. *Journal of Heuristics*, ročník 19, č. 6, 2013: s. 881–915.

[41] Jakobsen, T.: A Fast Method for the Cryptanalysis of Substitution Ciphers. *Cryptologia*, ročník 19, č. 3, 1995: s. 265–274.

[42] Jaskowiak, P. A.; Campello, R. J.; Costa, I. G.: On the selection of appropriate distances for gene expression data clustering. *BMC bioinformatics*, ročník 15, č. Suppl 2, 2014: str. S2.

[43] Kahn, D.: *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Scribner, 1996.

[44] King, J. C.; Bahler, D. R.: An Algorithmic Solution of Sequential Homophonic Ciphers. *Cryptologia*, ročník 17:2, 1993: s. 148–165.

[45] King, J. C.; Bahler, D. R.: A Framework for the Study of Homophonic Ciphers in Classical Encryption And Genetic Systems. *Cryptologia*, ročník 17, č. 1, 1993: s. 45–54.

[46] Kota, L.; Jármai, K.: Preliminary studies on the fixed destination MMTSP solved by discrete firefly algorithm. *Advanced Logistic systems*, ročník 7, č. 2, 2013: s. 95–102.

[47] Kraviarová, M.: Entropia a súčasná slovenská literárna tvorba. *Genologické a medziliterárne štúdie 6. Genologické konfrontácie.*, 2012: s. 145 – 152, [cit: 2015-08-11], Dostupné na internete: <`http://www.ff.unipo.sk/kraviarova/publikacie.html`>.

[48] Kraviarová, M.: Entropia ako nástroj na exaktné skúmanie literárneho diela. *Genologické a medziliterárne štúdie 7. Teória a interpretácia umeleckého textu.*, 2012: s. 59 – 88, [cit: 2015-08-11], Dostupné na internete: <`http://www.ff.unipo.sk/kraviarova/publikacie.html`>.

[49] Kubáček, L.: Confidence limits for proportions of linguistic entities. *Journal of Quantitative Linguistics*, ročník 1, č. 1, 1994: s. 56–61.

[50] Kullback, S.: Statistical Methods in Cryptanalysis. *Aegean Park Press*, 1976: str. 206.

[51] Kullback, S.: *Information Theory And Statistics*. Dover Pubns, 1997.

[52] Kvasnička, V.; Pospíchal, J.; Tiňo, P.: *Evolučné algoritmy*. Vydavateľstvo STU, Bratislava, 2000.

[53] Levitin, A. V.: *Introduction to the Design and Analysis of Algorithms*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006, ISBN 0321358287.

[54] Lin, J.: Divergence measures based on the Shannon entropy. *IEEE Transactions on Information theory*, ročník 37, 1991: s. 145–151.

[55] Luke, S.: Essentials of Metaheuristics. 2009, [cit: 2013-04-22], Dostupné na internete: <`http://cs.gmu.edu/~sean/book/metaheuristics/`>.

[56] Lyons, J.: Practical Cryptography. [cit: 2015-08-07], Dostupné na internete: <http://www.practicalcryptography.com/cryptanalysis/text-characterisation/chi-squared-statistic/>.

[57] Maison de la Recherche, Université Toulouse: Wikipedia FR2008 Corpus. [cit: 2016-11-09], Dostupné na internete: <http://redac.univ-tlse2.fr/corpora/wikipedia_en.html>.

[58] Malek, R.: Global Optimization through Meta-Heuristic Collaboration in a Multi-Agent System.

[59] Malek, R.: An agent-based hyper-heuristic approach to combinatorial optimization problems. In *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, ročník 3, oct. 2010, s. 428 –434.

[60] Mandl, P.: *Pravděpodobnostní dynamické modely*. Academia Praha, 1985.

[61] Matthews, R. A. J.: The Use of Genetic Algorithms in Cryptanalysis. *Cryptologia*, ročník 17, č. 2, 1993: s. 187–201.

[62] Meloun, M.; Militký, J.: *Kompendium statistického zpracování dat*. Academia, 2002.

[63] Menezes, A. J.; van Oorschot, P. C.; Vanstone, S. A.: *Handbook of Applied Cryptography*. CRC Press, 1996.

[64] Mirjalili, S.; Mirjalili, S. M.; Lewis, A.: Grey Wolf Optimizer. *Adv. Eng. Softw.*, ročník 69, Marec 2014: s. 46–61, ISSN 0965-9978.

[65] Mírka, J.: Dosud nevyluštěný dopis českého pobělohorského emigranta Karla Rabenhaupta ze Suché z doby třicetileté války. *CryptoWorld*, ročník 7-8, 2013.

[66] Mírka, J.: Ukázky nomenklátorů. 2013, [cit: 2013-12-07], Dostupné na internete: < http://soutez2013.crypto-world.info/pribeh/napoveda.pdf>.

[67] Navarro, G.: A Guided Tour to Approximate String Matching. *ACM Comput. Surv.*, ročník 33, č. 1, Marec 2001: s. 31–88, ISSN 0360-0300.

[68] Ólafsson, S.: Metaheuristics. *Handbooks in Operations Research and Management Science*, ročník 13, 2006: s. 633–654.

[69] Olson, E.: Robust Dictionary Attack of Short Simple Substitution Ciphers. *Cryptologia*, ročník 31, č. 4, Október 2007: s. 332–342, ISSN 0161-1194.

[70] Oranchak, D.: Evolutionary Algorithm for Decryption of Monoalphabetic Homophonic Substitution Ciphers Encoded As Constraint Satisfaction Problems. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, New York, NY, USA: ACM, 2008, ISBN 978-1-60558-130-9, s. 1717–1718.

[71] Ozcan, E.; Bilgin, B.; Korkmaz, E. E.: A Comprehensive Analysis of Hyper-heuristics. *Intelligent Data Analysis*, ročník 12, č. 1, 2008: s. 3–23.

[72] Pelling, N.: Gentlemen's Cipher. [cit: 2016-10-16], Dostupné na internete: <http://ciphermysteries.com/other-ciphers/the-gentlemens-cipher>.

[73] Perone, C. S.: Machine Learning : Cosine Similarity for Vector Space Models (Part III). [cit: 2015-08-09], Dostupné na internete: <http://blog.christianperone.com/2013/09/>.

[74] Pit-Claudel, C.: An experimental estimation of the entropy of English, in 50 lines of Python code. [cit: 2015-08-16], Dostupné na internete: <http://pit-claudel.fr/clement/>.

[75] Plesník, J.: *Grafové Algoritmy*. Vydavateľstvo SAV, 1983.

[76] Raddum, H.; Sýs, M.: The Zodiac killer ciphers. *Tatra Mountains Mathematical Publications*, ročník 45, 2010: s. 75 – 91.

[77] Rajaraman, A.; Ullman, J. D.: *Mining of Massive Datasets*. New York, NY, USA: Cambridge University Press, 2011.

[78] Ravi, S.; Knight, K.: Attacking Decipherment Problems Optimally with Low-Order N-gram Models. In *EMNLP*, ACL, 2008, s. 812–819.

[79] Reeves, C. R.: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, kapitola Fitness Landscapes. Boston, MA: Springer US, 2005, ISBN 978-0-387-28356-2, s. 587–610.

[80] Reid, W.: *The cipher dispatches*. Library of Congress, 1879, [cit: 2016-10-16], Dostupné na internete: <https://archive.org/stream/cipherdispatches01reid#page/42/mode/2up>.

[81] Rimarčík, M.: Chi-kvadrát test dobrej zhody. [cit: 2015-08-09], Dostupné na internete: <http://rimarcik.com/navigator/chi.html>.

[82] Rimarčík, M.: Zhluková analýza. [cit: 2015-08-09], Dostupné na internete: <http://rimarcik.com/navigator/ca.html>.

[83] Russell, M. D.; Clark, J. A.; Stepney, S.: Making the Most of Two Heuristics: Breaking Transposition Ciphers with Ants. *CEC 03*, ročník 4, 2003: s. 2653 – 2658.

[84] Savev, S.: Cosine Similarity Part 1: The Basics - Algorithms for Big Data. 2015, [cit: 2015-08-09], Dostupné na internete: <http://stefansavev.com/blog/cosine-similarity/>.

[85] Sayadi, M.; Ramezanian, R.; Ghaffari-Nasab, N.: A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, ročník 1, č. 1, 2010: s. 1–10.

[86] Schmeh, K.: Who can break these encrypted telegrams from 1876? 2016, [cit: 2016-10-16], Dostupné na internete: <`http://scienceblogs.de/klausis-krypto-kolumne/2016/08/11/who-can-break-these-encrypted-telegrams-from-1876/`>.

[87] Sebag-Montefiore, H.: *Enigma: Bitva o kód*. B4U, 2012, ISBN 9788087222096.

[88] Sekaj, I.: *Evolučné výpočty a ich využitie v praxi*. Vydavateľstvo IRIS, 2005.

[89] Sekaj, I.; Oravec, M.: Paralelné evolučné algoritmy. *Umelá inteligencia a kognitívna veda III*, 2011: s. 243 – 267.

[90] Serafino, L.: Between theory and practice: guidelines for an optimization scheme with genetic algorithms - Part I: single-objective continuous global optimization. *CoRR*, 2011.

[91] Shannon, C. E.: A Mathematical Theory of Communication. *The Bell System Technical Journal*, ročník 27, č. 3, 1948: s. 379–423.

[92] Shannon, C. E.: Communication Theory of Secrecy Systems. *Bell System Technical Journal, Vol 28, pp. 656–715*, 1949.

[93] Shannon, C. E.: Prediction and Entropy of Printed English. *Bell System Technical Journal*, ročník 30, 1951: s. 50–64.

[94] Shokhirev, N. V.: Distance and Metric. [cit: 2015-08-09], Dostupné na internete: <`http://www.shokhirev.com/nikolai/projects/popsci/metric/metric.html`>.

[95] Singh, A. P.; Pal, S. K.; Bhatia, M.: The Firefly Algorithm and Application in Cryptanalysis of Monoalphabetic Substitution Ciphers. *American Journal of Computer Science and Engineering Survey (AJCSES)*, ročník 1, č. 1, 2013: s. 33–52.

[96] Spillman, R.; Janssen, M.; Nelson, B.; aj.: Use of a Genetic Algorithm in the Cryptanalysis of Simple Substitution Cihers. *Cryptologia*, ročník 17, č. 1, 1993: s. 31–44.

[97] Státní oblastní archiv Třeboň: fond Rodinný archiv Buquoyů, sign. 223.223, inv.č. 730, položka 721/5-7 (1619, šifrovaný dopis Jaquota hraběti Buquoyovi).

[98] Štefánik, J.; Rusko, M.; Považanec, D.: Frekvencia slov, grafém, hlások a ďalších elementov slovenského jazyka. *Jazykovedný časopis*, ročník 50, č. 2, 1999: s. 81 – 93.

[99] T. Bagnall, G. P. M.; Rayward-Smith, V. J.: The cryptanalysis of a three rotor machine using a genetic algorithm. *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97), San Francisco, CA,*, 1997.

[100] Teahan, W.; Cleary, J.: Adaptive models of English text. 1997, (Working paper 97/30). Hamilton, New Zealand.

[101] Topsøe, F.; Fuglede, B.: *Jensen-Shannon Divergence and Hilbert space embedding.* IEEE Signal Processing Society, 2004, ISBN 0-7803-8280-3, str. 31.

[102] Uddin, M.; Youssef, A.: An Artificial Life Technique for the Cryptanalysis of Simple Substitution Ciphers. In *2006 Canadian Conference on Electrical and Computer Engineering*, 2006.

[103] Uddin, M. F.; Youssef, A. M.: Cryptanalysis of Simple Substitution Ciphers Using Particle Swarm Optimization. In *IEEE Congress on Evolutionary Computation*, 2006, s. 677–680.

[104] Urban, S.: Gentleman's Magazine. 4 1748, [cit: 2016-10-16], Dostupné na internete: <`http://www.bodley.ox.ac.uk/cgi-bin/ilej/image1.pl?item=page&seq=1&size=1&id=gm.1748.4.x.18.x.x.149`>.

[105] Vobbilisetty, R.; Troia, F. D.; Low, R. M.; aj.: Classic cryptanalysis using hidden Markov models. *to appear Cryptologia*: s. 1–28.

[106] Weise, T.: Global Optimization Algorithms Theory and Application. 2009, [cit: 2013-04-22], Dostupné na internete: <`http://www.it-weise.de/projects/book.pdf`>.

[107] Wikipedie, P.: Karel Bonaventura Buquoy. [cit: 2016-11-09], Dostupné na internete: <`https://cs.wikipedia.org/wiki/Karel_Bonaventura_Buquoy`>.

[108] Wimmer, G.; Altmann, G.; Hřebíček, L.; aj.: *Úvod do analýzy textov.* VEDA, 2003.

[109] Yang, X.: Firefly Algorithm, Stochastic Test Functions and Design Optimisation. *Int. J. Bio-Inspired Comput.*, ročník 2, č. 2, Marec 2010: s. 78–84, ISSN 1758-0366.

[110] Yang, X.-S.: A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, *Studies in Computational Intelligence*, ročník 284, Berlin, Heidelberg: Springer, 2010, ISBN 978-3-642-12537-9, s. 65–74.

[111] Yang, X.-S.: *Nature-Inspired Metaheuristic Algorithms: Second Edition.* Luniver Press, 2010, ISBN 9781905986286.

[112] Yang, X.-S.; Cui, Z.; Xiao, R.; aj.: *Swarm Intelligece and Bio-Inspired Computation: Theory and Applications*. Elsevier, 2013.

[113] Yang, X.-S.; Deb, S.: Cuckoo search via Lévy flights. In *World Congress on Nature & Biologically Inspired Computing (NaBIC 2009)*, IEEE Conference Publications, IEEE Publications, December 2009, s. 210–214.

[114] Yang, X.-S.; Deb, S.: Eagle Strategy Using Lévy Walk and Firefly Algorithms for Stochastic Optimization. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, *Studies in Computational Intelligence*, ročník 284, Berlin, Heidelberg: Springer, 2010, ISBN 978-3-642-12537-9, s. 101–111.

[115] Yu, S.; Yang, S.; Su, S.: Self-Adaptive Step Firefly Algorithm. *Journal of Applied Mathematics*, 2013.

[116] Zodiologist.com: [cit: 2013-04-22], Dostupné na internete: <`http://www.zodiologists.com/`>.