Dokumentace k MH\_decipher

Obsah

[Funkce get\_bigrams(text) 3](#_Toc200905871)

[Funkce substitute\_encrypt 4](#_Toc200905872)

[Funkce substitute\_decrypt 5](#_Toc200905873)

[Popis algoritmu prolom\_substitute – dešifrování substituční šifry bez klíče pomocí simulovaného žíhání a bigramové analýzy 6](#_Toc200905874)

[Závěr 8](#_Toc200905875)

## Funkce get\_bigrams(text)

**Popis:**

Funkce get\_bigrams analyzuje vstupní text a aktualizuje dodaný slovník frekvencí bigramů (dvoupísmenných sekvencí). Bigram je definován jako dvojice po sobě jdoucích znaků v řetězci. Pro každý výskyt bigramu ve vstupním textu se v příslušném slovníku buď navýší počet výskytů, nebo se bigram přidá jako nový záznam s počáteční hodnotou 1.

Funkce je navržena pro práci s již předzpracovaným textem – tj. textem očištěným o diakritiku, speciální znaky a převedeným do jednotné znakové sady (např. verzálkami). To zajišťuje konzistenci výsledného modelu pro další jazykové zpracování nebo kryptanalýzu.

**Parametry:**

text (string):

Předzpracovaný textový řetězec, ze kterého jsou extrahovány bigramy. Očekává se, že text je již normalizovaný (např. bez interpunkce, v jednotném formátu znaků).

bigram\_dict (slovník typu str -> int):

Slovník uchovávající výskyty bigramů. Klíčem je dvoupísmenný řetězec, hodnotou počet výskytů dané kombinace. Funkce tento slovník mění „in-place“.

**Návratová hodnota:**

Vrací odkaz na aktualizovaný vstupní slovník obsahující frekvence všech bigramů nalezených ve vstupním textu.

**Příklad:**

Zpracování textu "ABAB" vytvoří následující bigramy:

"AB" – výskyt 2×

"BA" – výskyt 1×

Výsledný slovník bude:

{'AB': 2, 'BA': 1}

**Vhodné použití:**

Funkce je ideální pro analýzu jazykových modelů, frekvenční rozbory nebo přípravu trénovacích dat pro metody strojového učení či šifrovací algoritmy založené na pravděpodobnostním hodnocení textu.

## Funkce substitute\_encrypt

**Popis:**

Funkce substitute\_encrypt slouží pro šifrování textu pomocí substituční šifry, kde každý znak původní abecedy je nahrazen znakem z uživatelem definovaného klíče (šifrovací abecedy). Je navržena jako vstupní bod pro externí volání a deleguje vlastní logiku do funkce en\_decipher, přičemž nastavuje parametr encode=True.

**Parametry:**

* ciphertext (string):  
  Cesta k souboru obsahujícímu vstupní text, který má být zašifrován.
* key (string):  
  Cesta k souboru, ve kterém se nachází substituční abeceda – tedy permutace původní abecedy, která definuje způsob nahrazování znaků.
* output\_file (string, volitelně):  
  Název výstupního souboru, do kterého se zapíše zašifrovaný text. Výchozí hodnota je 'output.txt'.
* default\_alphabet (string, volitelně):  
  Řetězec reprezentující původní abecedu (např. 27 znaků A-Z a \_). Výchozí hodnota je 'ABCDEFGHIJKLMNOPQRSTUVWXYZ\_'.

**Návratová hodnota:**

* Funkce nevrací žádnou hodnotu. Výsledek šifrování se zapisuje přímo do souboru určeného parametrem output\_file.

**Chování:**

* Funkce načte šifrovací klíč ze souboru.
* Validuje délku klíče vůči očekávané abecedě.
* Provede substituci každého znaku podle mapování default\_alphabet -> key.
* Zapíše výsledek do určeného souboru.

## Funkce substitute\_decrypt

**Popis:**

Funkce substitute\_decrypt provádí dešifrování textu, který byl dříve zašifrován pomocí substituční metody. Používá stejnou techniku jako substitute\_encrypt, ale provádí inverzní mapování – tedy převod znaků z klíčové abecedy zpět na znaky výchozí abecedy. Funkce je určena pro použití ve skriptech nebo jiných modulech a deleguje veškerou logiku na en\_decipher s parametrem encode=False.

**Parametry:**

* ciphertext (string):  
  Cesta k souboru, který obsahuje šifrovaný text k dešifrování.
* key (string):  
  Cesta k souboru obsahujícímu šifrovací abecedu. Tato abeceda bude reverzně namapována zpět na původní znaky.
* output\_file (string, volitelně):  
  Název výstupního souboru, do kterého se uloží dešifrovaný výstup. Výchozí hodnota je 'output.txt'.
* default\_alphabet (string, volitelně):  
  Výchozí abeceda použitá při šifrování. Délka musí odpovídat délce klíče.

**Návratová hodnota:**

* Funkce nevrací žádnou hodnotu. Výstupní dešifrovaný text je zapsán do určeného výstupního souboru.

**Chování:**

* Načte zvolený klíč ze souboru.
* Ověří jeho platnost vůči výchozí abecedě.
* Provádí zpětné mapování key -> default\_alphabet.
* Výsledek uloží do specifikovaného souboru.

## Popis algoritmu prolom\_substitute – dešifrování substituční šifry bez klíče pomocí simulovaného žíhání a bigramové analýzy

Funkce prolom\_substitute implementuje statistický přístup k prolomení substituční šifry bez znalosti klíče. Klíčovým prvkem je využití bigramové matice, která reprezentuje pravděpodobnosti dvojic písmen v daném jazyce, čímž umožňuje vyhodnotit, jak dobře dešifrovaný text odpovídá přirozenému jazyku.

**Princip fungování algoritmu:**

1. **Vstupní parametry**:
   * text: šifrovaný text (ciphertext) určený k dešifrování.
   * TM\_ref: relativní pravděpodobnosti výskytu bigramů (bigramová matice).
   * iter\_count: počet iterací, během kterých bude algoritmus hledat nejlepší klíč.
   * alphabet: abeceda použitá pro šifru (standardně velká písmena a podtržítko).
   * start\_key (volitelný): počáteční klíč, pokud není zadán, algoritmus začne s náhodným klíčem.
2. **Inicializace**:
   * Pokud není zadán počáteční klíč, vytvoří se náhodná permutace abecedy.
   * Spočítají se logaritmické pravděpodobnosti bigramů, které slouží pro rychlé vyhodnocení "věrohodnosti" dešifrovaného textu.
   * Vyhodnotí se počáteční skóre plausibility (jak dobře klíč odpovídá statistice jazyka).
3. **Iterativní vylepšování klíče (simulated annealing)**:
   * V každé iteraci se provede náhodná drobná úprava klíče (prohození dvou písmen).
   * Dešifruje se text tímto kandidátním klíčem a spočítá se jeho plausibility skóre.
   * Pokud je skóre lepší než současné, klíč se přijme.
   * Pokud je skóre horší, klíč se může přijmout s určitou pravděpodobností závislou na tzv. „teplotě“ (implementováno pomocí exponenciální funkce). Tento mechanismus umožňuje uniknout z lokálních maxim.
   * Teplota postupně klesá (ochlazování), čímž se algoritmus zaměřuje na jemné dolaďování řešení.
4. **Resety a ukončení**:
   * Pokud po určitém počtu iterací není žádné zlepšení, klíč se resetuje na nejlepší doposud nalezený.
   * Po překročení předem daného počtu iterací bez zlepšení se algoritmus ukončí (early stop).
5. **Výstup**:
   * Nejlepší nalezený klíč.
   * Dešifrovaný text odpovídající tomuto klíči.

**Výhody a omezení:**

* **Výhody**:
  + Algoritmus nevyžaduje znalost původního klíče.
  + Efektivně využívá jazykové statistiky (bigramy), což výrazně zvyšuje šanci na úspěšné dešifrování.
  + Simulované žíhání pomáhá vyhnout se lokálním extrémům v prostoru klíčů.
* **Omezení**:
  + Výkon závisí na kvalitě a rozsahu bigramové matice.
  + Pro velmi krátké texty nemusí být statistika dostatečně významná.
  + Počet iterací a parametry „teploty“ je třeba vhodně nastavit pro dobrý kompromis mezi rychlostí a kvalitou výsledku.

## Závěr

Představený modul tvoří ucelený nástroj pro práci se substitučními šiframi, kombinující základní kryptografické operace se statistickou analýzou přirozeného jazyka. Funkce get\_bigrams zajišťuje extrakci bigramových frekvencí z předzpracovaného textu, což tvoří základ pro jazykové modelování. Na ni navazují metody substitute\_encrypt a substitute\_decrypt, které poskytují robustní a konfigurovatelné rozhraní pro šifrování i dešifrování textu pomocí libovolné substituční abecedy.

Nejvýznamnějším prvkem celého řešení je algoritmus prolom\_substitute, který umožňuje dešifrovat substituční šifru bez znalosti klíče. Využitím bigramové analýzy a optimalizační techniky simulovaného žíhání dosahuje překvapivě kvalitních výsledků i při absenci jakékoli počáteční znalosti původního obsahu. Tento přístup ukazuje sílu spojení jednoduché frekvenční statistiky s pravděpodobnostními metodami a poskytuje efektivní řešení pro základní kryptanalytické úlohy.

Celý modul tak nachází uplatnění nejen v oblasti výuky kryptografie, ale i v praktickém testování robustnosti šifrovacích metod nebo jako výchozí komponenta pro vývoj pokročilejších jazykově založených dešifrovacích systémů.

Pokud potřebuješ závěr zkrátit, přeformulovat do méně formálního tónu nebo přizpůsobit určitému účelu (např. součást školního projektu nebo technické dokumentace), dej vědět – upravím ho podle potřeby. A pokud mi chceš poskytnout i kód nebo projekt, rád ti pomůžu s doladěním i dalších částí.