Žaneta Grossová
xgross11
8. 4. 2023

Dokumentace

Afinitní šifra

Projekt č. 1
Kryptografie

Afinitní šifra

Afinitní šifra je jednoduchá substituční šifra, která se používá k šifrování zpráv. Její základní princip spočívá v lineární transformaci jednotlivých písmen v textu za pomocí dvou klíčů "a" a "b". Tyto klíče ovlivňují, jaké písmeno bude z původního textu zakódováno na konkrétní písmeno v zašifrovaném textu.

Šifrování

Celá rovnice pro zašifrování jednoho písmena je následující: **E(x) = (a * x + b) % 26**,
Kde "a" a "b" značí zadané klíče, "x" značí index znaku, který šifrujeme a "%" značí operaci modulo, přičemž klíč "a" může dle zadání být pouze prvočíslo s délkou abecedy "m" – tedy v našem případě může nabývat pouze hodnot 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23 a 25.

Dešifrování

Celá rovnice pro dešifrování jednoho písmena je následující: **D(x) = a−1(x – _b) % 26**,
Kde "a" a "b" značí zadané klíče, "x" značí index znaku, který šifrujeme a "%" značí operaci modulo, přičemž klíč "a" může dle zadání být pouze prvočíslo s délkou abecedy "m" – tedy v našem případě může nabývat pouze hodnot 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23 a 25.

Implementace

Program je psán bez využití objektové orientace, skládá se pouze z hlavní funkce *main()* a několika doplňkových funkcí. Nejdříve proběhne volání funkce *parse_parameters(),* která zpracuje vstupní parametry a uloží je do argumenty předaných polí. V případě, že není zadán parametr označující vstup nebo výstup ze souboru, název souboru je nahrazen prázdným řetězcem a vstupní text se bere ze zadaného parametru. Na základě hodnot názvu vstupního a výstupního souboru se následně v programu rozhoduje, zda jsou volány funkce *get_input_from_file()* a *write_output_to_file()*, které slouží pro práci se soubory, nebo zda se text bere nebo zapisuje na standardní výstup. Dle hodnoty přepínače je volána jedna z funkcí *decrypt(), encrypt()* nebo *hack()*, které slouží pro šifrování, dešifrování a nebo pro dešifrování bez znalosti klíče. Všechny využívají doplňkové funkce *get_order_number()* a *get_letter_from_order_number()*, které převádí předaný argument buď na číslo odpovídající indexu daného znaku v abecedě, který je stejný pro velké i malé písmeno, nebo z indexu odvodí a vrátí odpovídající znak abecedy. Šifrování a dešifrování se pak provádí pouze dosazením hodnot do rovnic odpovídajících dané operaci.

Detaily implementace dešifrování bez znalosti klíče

Funkci *hack()* je jako vstupní argument zadán pouze předaný zašifrovaný text. Jako první se spočítají výskyty všech znaků, které se následně uloží do struktury typu *map*, kde se počet výskytů namapuje na daný znak. V této struktuře se následně naleznou tři nejčastěji se vyskytující znaky.

Žaneta Grossová
xgross11
8. 4. 2023

Následuje jádro celé frekvenční analýzy – v několika *for* smyčkách se zkouší hodnoty klíčů "a" a "b" tak, že jsou do šifrovací rovnice se zkoušenou kombinací klíčů dosazeny postupně indexy nejčastějších písmen české abecedy a porovnává se jejich zašifrovaná hodnota s třemi nejčastějšími znaky v šifrovaném vstupním textu. Pokud je výsledek pravdivý pro první nejčastější znak, porovnává se daná kombinace klíčů v rovnici spolu s druhým a následně třetím nejčastějším znakem. Pokud je nalezena kombinace klíčů, která vyhovuje všem třem rovnicím, je jejich dvojice uložena do vektoru datových typů *pair*. Pro optimalizaci se využívá vlastnosti klíče "a", který může nabývat pouze několika hodnot, klíč "b" se pak zkouší pro rozsah 0 - 25, vzhledem k operaci modulo 26, která vrací zbytek po dělení danou hodnotou by se výsledek pro vyšší hodnoty než 25 shodoval s již zkoušenými nižšími hodnotami a výsledků by tak mohlo být nekonečně mnoho.

Následně je z vektoru vyhovujících klíčů nutné vybrat takovou dvojici, která má největší pravděpodobnost, že jejím dešifrováním vznikne smysluplný text. Testováním na kratších textech se v průběhu implementace ukázalo, že čtvrtým nejčastějším znakem může být často znak, který v českém jazyce není tolik frekventovaný, protože stačí, že se v textu objeví dvakrát, například pokud bude v textu anglický název šifry „Affine“, znak 'f' se může stát čtvrtým nejčastějším znakem i přestože je v českém jazyce jedním z nejméně frekventovaných. Proto se ukázal jako mnohem efektivnější opačný přístup - cyklením přes daný vektor klíčů počítat výskyty nejméně frekventovaných znaků českého jazyka. Písmena 'Q', 'W' a 'X' jsou tři nejméně frekventované znaky českého jazyka a pokud se nejedná o vysoce odborný text, jejich výskyt v běžném textu je velmi ojedinělý, často nulový. Ukázalo se, že dešifrováním špatnými hodnotami klíčů se ale v textu vyskytují velmi často, čehož bylo při implementaci využito. Vektor vhodných kandidátů na klíče je procházen ve smyčce, znaky 'Q', 'W' a 'X' jsou zašifrovány zkoušenou kombinací klíčů a následně se prochází vstupní zašifrovaný text, ve kterém se počítá výskyt těchto tří znaků dohromady. Počet je namapován na číslo indexu ve vektoru klíčů, které byly zkoušeny, takže následně stačí v uložené mapě najít index s nejmenším počtem výskytů, díky kterému získáme kombinaci klíčů, která má největší pravděpodobnost, že jejím dešifrováním vznikne smysluplný text. Tento postup se ukázal jako velice efektivní a často funguje i na krátké texty, které by jinak pouhým porovnáváním nejčastějších znaků nebyly prolomeny.