Descrierea Aplicației **Naevia**

Apostolescu Ștefan
Băjan Ionuț-Mihăiță
Iosif George-Andrei

01/02/2021

Tabelă de Conținut

1 Introducere

Aplicația dezvoltată ca temă pentru cursul de "*Inteligență Artificială*" poartă numele de **Naevia**. Aceasta își propune folosirea tehnicii *Hill Climbing* pentru optimizarea procesului de criptanaliza a unor cifruri clasice. În același timp, prin intermediul ei, se pot cripta texte cu ajutorul cifrurilor implementate și face comparații între abordarea bazată pe forță brută (engl. "*brute-force*") și cea optimizată, prin intermediul unui grafic ce ilustrează dificultatea cu care o anumită abordare tinde spre o soluție.

De menționat este principalul dezavantaj al folosirii *Hill Climbing* pentru criptanaliză. Acesta constă în faptul că tehnica asigură numai găsirea unei soluții locale, care poate coincide cu textul în clar inițial, folosit în procesul de criptare. Din acest motiv, pentru cazurile în care soluția returnată nu este cea validă (lucru determinat, de exemplu, prin intermediul sensului avut de text) se recomandă rularea repetată pentru că, la fiecare rulare, punctul de plecare din spațiul de valori (în cazul de față, cheia încercată în procesul de decriptare) este unul diferit.

2 Noțiuni Teoretice

2.1 Cifruri Clasice

2.1.1 Caesar

Cifrul lui Caesar este un sistem de cifrare monoalfabetic pentru care textul în clar este construit din literele alfabetului latin (de la A la Z) și din cheia de cifrare, reprezentată de un număr întreg k $\in$ { 0, ..., 25}.

$$E_i(x) = (M_i + K - 65)\, mod\ 26 + 65,$$

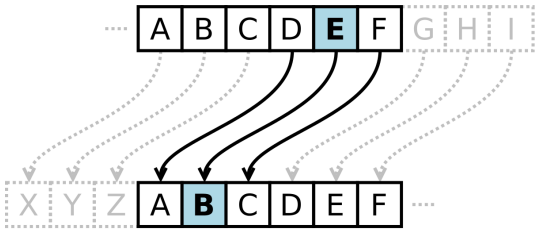
$$E - mesajul\ criptat; M - mesajul\ în\ clar, K - cheia\ de\ criptare, i = \overline{1, L}, \quad unde,$$

$$L - lungimea\ mesajul\ în\ clar$$

Imagine 1: Criptarea unui text cu ajutorul cifrului Caesar

$$M_i(x) = (D_i - K - 65)\, mod\ 26 + 65, \quad D - mesajul\ criptat$$

Imagine 2: Decriptarea unui text cu ajutorul cifrului Caesar



Imagine 3: Transformarea alfabetului în cadrul cifrului Caesar

După cum se observă, acest cifru este unul simplu întrucât, în procesul de criptare, fiecare caracter din mesajul inițial este deplasat circular în cadrul alfabetului. În acest sens, complexitatea algoritului de cifrare este direct proporțională cu lungimea alfabetului. În cazul alfabetului în limba engleză, sunt necesare 26 de încercări, folosind un atac cu forță brută, pentru a afla mesajul inițial.

2.1.2 Vigenère

Cifrul lui Vigenère este o metodă de criptare care folosește o serie de cifruri Caesar diferite, bazate pe literele unui cuvânt-cheie. Este o formă simplă de substituție polialfabetică. În acest cifru, cheia de criptare este reprezentată de o parolă (de regulă, un cuvânt). Dacă dimensiunea cheii de criptare este mai mică decât cea a mesajului, se repetă cheia până la finalizarea codificării mesajului.

$$E_i(x) = (M_i + K_j)mod\ 26 + 65\ ,$$

$$E - mesajul\ criptat; M - mesajul\ în\ clar\ , K_j - cheia\ de\ criptare, i = \overline{1, L}, j = \overline{1, l}\ \ , l - lungimea\ cheii\ , L - lungimea\ mesajul\ în\ clar$$

Imagine 4: Criptarea unui text cu ajutorul cifrului Vigenère

$$E_i(x) = (M_i - K_j)mod\ 26 + 65$$

Imagine 5: Decriptarea unui text cu ajutorul cifrului Vigenère

Complexitatea acestui cifru este dată de lungimea cheii folosite. Presupând că un atacator ghicește lungimea parolei, spargerea cifrului Vigenère poate fi tratată ca în cazul cifrului Caesar. Pentru a afla lungimea chei se pot folosi următoarele teste:

- Kasiski, ce profită de faptul că toate cuvintele repetate sunt, din întâmplare, uneori criptate folosind aceleași litere-cheie, ceea ce duce la grupuri repetate în textul cifrat; și

- Friedman, ce se bazează pe indicele de coincidență, care măsoară denivelarea frecvențelor literelor cifrate pentru a sparge cifrul.

2.1.3 De Substituție

Operația de criptare în cadrul cifrului de substituție se bazează pe o corespondență biunivocă între alfabetul clar și cel cifrat. Cheia de criptare în acest cifru este reprezentată de o permutare a dicționarului folosit la generarea mesajului inițial. Permutarea poate fi aleatoare sau pseudoaleatoare.

$$E_i(x) = K_i(M_i)\,,$$

$M_i - \textit{literă din mesajul inițial},\ K_i - \textit{simbol al noului dicționar utilizat}$

Imagine 6: Criptarea unui text cu ajutorul cifrului de substituție

$$D_i(x) = C_i(K_i)$$

Imagine 7: Decriptarea unui text cu ajutorul cifrului de substituție

La o primă vedere, pentru spargerea acestui cifru printr-un atac prin forță brută, avem un spațiu total al tutoror cheilor posibile de aproximativ 26 factorial (număr reprezentat pe aproximativ 88 de biți). Însă de considerat este faptul că, în practică, ultimele litere ale alfabetului (care sunt în mare parte de frecvență joasă) tind să rămână la sfârșit. Acest lucru reduce semnificativ timpul de spargere a cheii de criptare.

2.2 *Hill Climbing*

Hill Climbing este un algoritm de căutare optimizată a soluțiilor unei probleme care se poate rezolva prin minimizare. Acesta folosește o funcție de euristică, precum și o funcție de generare de stări, amândouă specifice fiecărei probleme în parte. Algoritmul pornește de la o stare inițială și, prin iterații succesive, ajunge la un punct de minim local.

3 Folosirea *Hill Climbing* în Contextul Criptanalizei

Hill Climbing poate fi aplicat în cripanaliza clasică, bazată pe substituție, pentru spargerea cheii de criptare.

Algoritmul pleacă de la o cheie inițială, dată sau alesă aleator. Cu ajutorul funcției de generare de stări, se derivă o mulțime de alte chei. Textul este decriptat cu ajutorul acestor chei și, pentru fiecare variantă a textului decriptat, se calculează o metrică. Din mulțimea de chei generate anterior se alege cheia pentru care textul decriptat are euristica (metrica stabilită) cea mai mică. Euristica este calculată în așa fel încât să tindă spre 0 atunci când textul decriptat este în limba engleză.

Procesele de derivare, decriptare și de calculare a euristicii se repetă până când nu se mai găsește o cheie cu euristică mai mică decât cheia curentă.

3.1 Funcția de Generare de Stări

În cazul criptografiei clasice, bazată pe substituție, funcția de generare de stări reprezintă funcția de derivare de chei. Aceasta primește ca parametru o cheie inițială și returnează o mulțime de chei derivate din aceasta. Cheile derivate sunt obținute aplicând permutări literelor cheii inițiale, cât și înlocuind aleator litere acesteia. Mulțimea obținută conține chei unice atât la nivel local (toate cheile generate într-o iterație) cât și la nivel global (toate cheile deja generate).

3.2 Funcția Euristică

Funcția euristică își propune să determine asemănarea textului descifrat cu unul în limba engleză. Aceasta se poate verifica numărând frecvența de apariție a literelor sau a grupurilor de litere din textul decriptat și comparând frecvențele cu cele determinate empiric, ale limbii engleze. Cu cât textul se apropie de unul în limba engleză, cu atât mai mult va scădea valoarea euristicii.

$$P(\text{“abcd”}) = \frac{nr(\text{abcd})}{n}$$ unde nr (“abcd”) reprezintă numărul total de apariții a secvenței “abcd” în text și **n** numărul total de secvențe de lungime stabilită din text.

Imagine 8: Formula pentru calcularea frecvenței unei grupări de 4 litere

$$P(\text{“abcde”}) = P(\text{“abcd”}) \times P(\text{“bcde”})$$

Imagine 9: Formula pentru calcularea frecvenței unei grupări de 5 litere

Pentru un text de lungime medie, calculul tuturor frecvențelor va presupune multe înmulțiri și vor apărea erori numerice de calcul. Pentru a evita acest lucru, se va calcula logaritmul fiecărei probabilități astfel:

$$log\big(P(\text{abcde})\big) = log\big(P(\text{abcd})\big) + log\big(P(\text{bcde})\big)$$

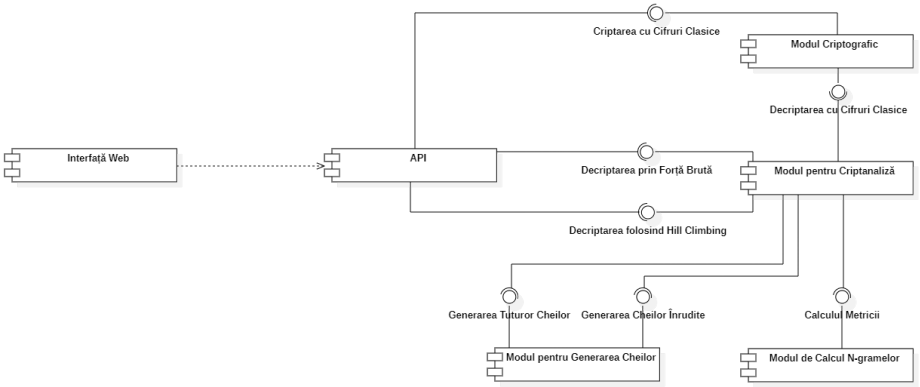
folosind identiatea: $log(a * b) = log(a) + log(b)$

Practic, calcularea euristicii pentru un text se rezumă la calcularea sumei logaritmilor frecvențelor de apariție a grupurilor de n litere, unde n este număr natural mai mare decât 1. Pentru aplicația dezvoltată, valoarea lui n este egală cu 4.

Dat fiind spațiul uriaș de căutare în cazul substituției pentru o cheie de 26 de caractere, anume 26^{26}, există o probabilitate mare ca algoritmul să nu găsească din prima iterație minimul global, ci să se oprească la un minim local. De aceea este recomandată rularea de mai multe ori a algoritmului (și folosirea cheii găsite cel mai des) sau analiza manuală a textului decriptat.

Prin metode experimentale, s-a estimat probabilitatea de găsirii a cheii corecte ca fiind $\frac{2}{3}$ din numărul total de încercări. Această probabilitate variază în funcție de lungimea textului criptat. Cu cat acesta este mai lung, cu atat mai mult vor varia euristicile pentru diferitele chei, iar cheia corectă va fi găsită mai ușor.

4 Arhitectura Aplicației



Imagine 10: Diagrama de Componente a Aplicației

5 Modul de Utilizare al Aplicației

Utilizatorii aplicației pot interacționa cu aceasta prin intermediul unei interfețe web, compusă dintr-o singură pagină (engl. "*single-page application*") în care conținutul computațiilor efectuate de serverul web sunt incluse în manieră dinamică în ea, fără a fi necesară o reîncărcare completă.

O interacțiune uzuală cu aplicația conține următorii pași:

- accesarea aplicației cu ajutorul unui *browser* web;
- selectarea operațiunii ce se dorește efectuată (criptare sau decriptare);
- în cazul în care criptarea este aleasă:
  - setarea textului în clar;
  - alegerea unui cifru disponibil (Caesar, Vigenère sau cifru de substituție);
  - setarea parolei pentru cifru (un număr pentru Caesar, un șir de caractere pentru Vigenère sau o permutare a alfabetului pentru cifrul de substituție);
  - apăsarea butonului de trimitere a cererii de criptare;
  - așteptarea primirii unui răspuns de la serverul web; și
  - preluarea textului criptat din câmpul specific.
- în cazul în care decriptarea este aleasă:
  - setarea textului criptat;
  - alegerea unui abordări disponibile pentru decriptare (prin *brute-force* sau cu ajutorul tehnicii de optimizare *Hill Climbing*);
  - apăsarea butonului de trimitere a cererii de decriptare;
  - așteptarea primisii unui răspuns de la serverul web;
  - preluarea textului decriptat din câmpul specific; și
  - observarea în grafic a modului în care funcția euristică a tins către soluția returnată.

6 Concluzii

În concluzie, aplicația dezvoltată dovedește faptul că tehnica de optimizare *Hill Climbing* poate fi utilizată cu succes în procesul de criptanaliza a unor cifruri clasice. Deși tehnica asigură numai un minim local, deci numai un text care ar putea fi cel inițial, experimentele realizate au demonstrat o eficiență ridicată.