

INFO002 : cryptologie TD 2 : un peu de probabilité – indice de coïncidence

Pierre Hyvernât
Laboratoire de mathématiques de l'université de Savoie
bâtiment Chablais, bureau 17, poste : 94 22
email : Pierre.Hyvernât@univ-smb.fr
www : <http://www.lama.univ-smb.fr/~hyvernât/>

Exercice 1 : indice de coïncidence

L'*indice de coïncidence* d'un texte (c_i) est défini comme la probabilité que deux caractères c_i et c_j ($i \neq j$) pris au hasard soient égaux.

Question 1.

- Quel est l'indice de coïncidence de `abcdefghijklmnopqrstuvwxy` ?
- Et pour `aabbccddeeffgghhijjkkllmmnnnooppqrrssttuuvvwxxyyzz` ?
- Quel est l'indice de coïncidence d'un long texte aléatoire utilisant n symboles différents ?

Question 2.

- Que pouvez-vous dire de l'indice de coïncidence d'une *permutation* d'un texte ?
- Que pouvez-vous dire de l'indice de coïncidence d'un chiffrement monoalphabétique d'un texte ?

Question 3. On se donne un texte (c_i) de longueur n , et on note n_a le nombre d'occurrences du caractère **a**, n_b le nombre d'occurrences du caractère **b**, etc.

Donnez une formule qui permet de calculer son indice de coïncidence.

Question 4. Pourquoi l'indice de coïncidence d'un texte clair est-il plus élevé que $1/26 \approx 0.038$?

Question 5. Comment peut-on utiliser l'indice de coïncidence pour essayer de deviner la taille de la clé d'un message chiffré *polyalphabétiquement* ?

Exercice 2 : recherche exhaustive

Question 1. On suppose que le cardinal des clés est N , et que les clés sont générées uniformément (chaque clé est donc tirée avec probabilité $1/N$).

On suppose connu un texte clair m et son chiffré c . On recherche une clé k telle que $D_k(c) = m$.

Quelle est l'espérance du nombre de déchiffrements à effectuer avant de trouver une telle clé lors d'une recherche exhaustive ?

Indice : les événements "la bonne clé est la première", "la bonne clé est la deuxième", "la bonne clé est la troisième", ... sont tous équiprobables.

Exercice 3 : attaques sur le "bloc note à usage multiple"

Le "one time pad" a la propriété du secret parfait. Le "two times pad", où la clé est réutilisée (une ou plusieurs fois) n'est plus sûr.

Question 1. Expliquez pourquoi une attaque par recherche exhaustive de la clé n'est pas possible pour le bloc note à usage unique.

Voici un exemple d'utilisation particulièrement mauvaise du “ t times pad” ($t \geq 2$). Dans la suite, on suppose que

$$\mathcal{M} = \mathcal{K} = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{g}, \mathbf{h}, \mathbf{i}, \mathbf{j}, \mathbf{k}, \mathbf{l}, \mathbf{m}, \mathbf{n}, \mathbf{o}, \mathbf{p}, \mathbf{q}, \mathbf{r}, \mathbf{s}, \mathbf{t}, \mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{x}, \mathbf{y}, \mathbf{z}, \sqcup\}^n$$

($\sqcup$ représente l'espace)

Le chiffrement s'effectue par le XOR bit à bit sur la suite des codes ASCII ($\mathbf{a} = 01100001$ (97), $\mathbf{b} = 01100010$ (98), ... $\mathbf{z} = 01111010$ (122), et $\sqcup = 00100000$ (32))

Question 2. Que pouvez-vous dire du XOR entre 2 lettres par rapport au XOR entre 1 lettre et $\sqcup$?

Question 3. Décrivez une attaque possible pour retrouver la clé si vous disposez de $t > 2$ textes chiffrés (de taille n) avec la même clé (elle aussi de taille n).

Question 4. Que pensez-vous du cas $t = 2$?

Question 5. Implémentez un petit programme pour craquer (de manière interactive) des textes chiffrés de cette manière (avec $t > 2$). Vous trouverez des exemples de fichiers chiffrés sur ma page web.

Note :

- pour récupérer un tableau d'octets correspondant à un fichier, en Python, vous pouvez utiliser `B = bytearray(open(fichier, mode="rb").read())` ;
- pour transformer un octet en caractère ASCII, vous pouvez utiliser la fonction `chr`, et pour transformer un caractère ASCII en octet, la fonction `ord` ;
- le XOR, en Python (C, Java, Rust, Go, ...), s'écrit avec un “`^`”.