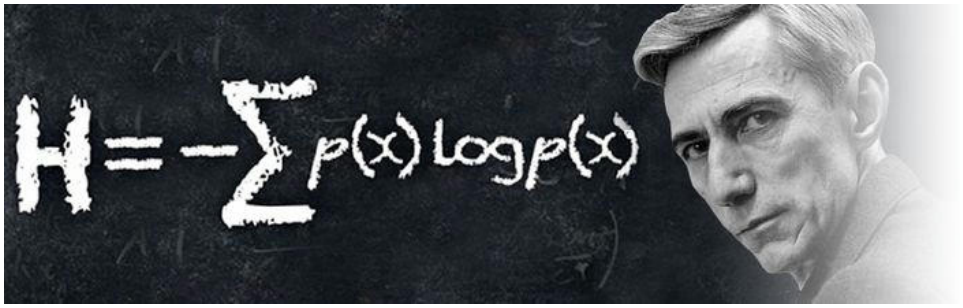
THÉORIE DE L'INFORMATION DE CLAUDE E. SHANNON

LASSERRE JORIS ET TRISTAN PETIT



~

Créé le 26/03/2024

Sommaire

1. Introduction

Nous utilisons tous les jours nos ordinateurs pour nous connecter à Internet, s'inscrire, se connecter sur nos sites de streaming préférés ou encore s'envoyer de lourds fichiers zippés. Si tout cela est possible, c'est grâce à la théorie de l'information élaborée par Claude E. Shannon qui a servi de fondement pour un grand nombre de travaux. Claude Shannon, mathématicien et ingénieur américain, a été le premier, en 1948 à définir un cadre mathématique pour la transmission d'informations à travers des canaux alors que tout ce que nous savions se basait sur des procédés empiriques.
Cette théorie a permis de trouver un moyen de mesurer et quantifier l'information grâce au bit `(Voir section choix des unités de mesure)`.
Dans ce dossier nous allons découvrir plusieurs principes développés par Shannon tels que l'entropie, la redondance, le bruit, les codes correcteurs d'erreur etc. Nous examinerons ensuite ces principes qui ont été appliqués dans des `domaines divers` comme les télécommunications, la compression de données ou encore la cryptographie.

2. Fondements de la théorie de l'information

Pour comprendre la théorie de l'information nous avons besoin de définir un certain nombre de notions qui nous serons indispensables tout au long de cet article.

2.1. **Choix des unités de mesure.**

Notons *p(A)* la probabilité d'un évènement *A* et *h(A)* son information. Un évènement rare apporte plus d'information s'il se produit qu'un évènement courant. Donc si la probabilité *p(A)* de A augmente, la quantité d'information *h(A)* diminue et inversement. On peut donc exprimer l'information *h* comme ceci

$$h(A) = f\left(\frac{1}{p(A)}\right)$$

Cette fonction f doit remplir les conditions suivantes :

- f est croissant
- si p(A) = 1 alors h(A) = 0 (un évènement certain n'apporte aucune information)
- f(p1 . p2) = f(p1) + f(p2) (l'information apportée par 2 évènements indépendants est la somme de l'information de chaque évènement)

La fonction logarithme est la seule fonction qui remplit ces conditions.

On obtient donc la formule de l'information $h(A) = -\log P(A)$

Shannon a ensuite décidé d'utiliser le logarithme en base 2 car le résultat de celui-ci est 1 ou 0, ces valeurs, il les a nommé *bits* en référence au terme "binary digits".

Le *bit* est la plus petite quantité d'information dans un message, elle ne contient que 0 ou 1. Elle constitue donc l'unité de base de l'information.

2.2. **Entropie.**

Pour comprendre ce qu'est l'entropie nous allons partir de plusieurs définitions :

Définition [1] : L'entropie d'une variable aléatoire est une mesure quantitative de l'incertitude (ou, alternativement, de la quantité d'information) associée aux valeurs prises par la variable aléatoire.

Définition [2] : L'entropie d'une distribution c'est l'information en nombre de bits contenu en moyenne par message lorsque ceux-ci sont pris dans une distribution donnée.

Nous calculons l'entropie en analysant les probabilités présentes dans un ensemble de données. Si toutes les valeurs présentent la même probabilité, cela entraîne une grande incertitude et une entropie maximale. Au contraire, si une des valeurs que nous avons obtenues a une probabilité plus forte que les autres, l'entropie est réduite car il y a moins d'incertitude dans la réalisation de la variable aléatoire.

Prenons le cas d'un lancer de pièce non truquée, il y a une probabilité de 0.5 de faire face ou pile. Chaque événement ayant la même probabilité de se réaliser, l'incertitude est maximale donc l'entropie est maximale.
Au contraire, si nous prenons un dé pipé dont les probabilités sont les suivantes :

Face du dé	**Probabilité**
1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.5

Les chances d'obtenir le chiffre 6 sont plus élevées donc l'entropie est faible, il y a peu d'incertitude.

La formule de l'entropie souvent notée H(X) est la suivante: $H(X) = -\sum_{i=1}^{n} P_i \log_2 P_i$
où X est une variable aléatoire, P_i représente la probabilité de l'événement i et n est le nombre total d'événements possibles Les valeurs de l'entropie sont toujours nulles ou positives. Une valeur nulle indique que le résultat est prévisible, qu'il n'y a pas d'incertitude tandis qu'une valeur élevée indique une incertitude élevée.

2.3. **Redondance.**

Dans la théorie de l'information, la redondance correspond à la quantité d'information répétées ou inutiles, c'est-à-dire la différence entre la quantité de bits nécessaire pour transmettre un message et la quantité de bits réellement utilisée.
Nous allons voir dans la partie application de la théorie de l'information que la redondance est utilisée pour corriger les erreurs lors de la transmission en rajoutant des bits ou encore lors de la compression de données où nous allons supprimer les informations redondantes.

La formule de la redondance est $1 - H(X)$ où H(X) est l'entropie.
On remarque que que si l'entropie est élevée, la redondance sera faible. A l'inversement, une entropie faible engendre une redondance élevée.

2.4. **Canal discret.**

Shannon base sa théorie sur la transmission d'informations à travers un canal discret. Un canal discret est un canal de communication (par exemple un télégraphe ou le WIFI) qui transmet un nombre fini de symbole. Il définit ainsi que les canaux ont une capacité :

$$C = \lim_{T \to \infty} \frac{\log N(T)}{T}$$

où N(T) est le nombre de signaux autorisés de durée T.

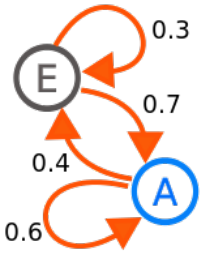
Cette capacité n'indique pas que le canal transmettra toujours à cette vitesse, il indique seulement que la vitesse maximale de ce canal est C. Dans son article Shannon parle essentiellement de canal discret sans bruit ou avec bruit. Un canal sans bruit désigne un canal où l'information de départ et celle arrivée en sortie sont les mêmes, sans erreur de transmission ni de perturbation, sa probabilité de transmission est de 1. Au contraire, un canal bruité est un canal dont l'information est soumis à des perturbations qui peuvent être de sources variées (par exemple interférence électrique). L'information n'est donc plus la même en entrée qu'en sortie et sa probabilité est inférieure à 1, plus le bruit est fort et plus la probabilité est faible.
Il représente le bruit par la formule ci-dessous :
$C = W.\log_2\big(1 + \big(\frac{P}{N}\big)\big)$ bits/s où W est la largeur de bande (en Hertz) et $\frac{P}{N}$ le rapport signal à bruit présent dans la transmission, P désignant la puissance du signal et N la puissance du bruit (Noise).

2.5. **Processus stochastique.**

Dans son article Shannon considère qu'une source discrète peut être vue sous la forme d'un modèle mathématique appelé processus stochastique qui émet des symboles les uns après les autres selon une certaine distribution de probabilité.

Les processus stochastiques utilisés sont des processus de Markov, dont la principale propriété, est que la probabilité qu'un système passe à un état futur dépend uniquement de son présent et non de son passé (ensemble des états précédents). Ces états sont les

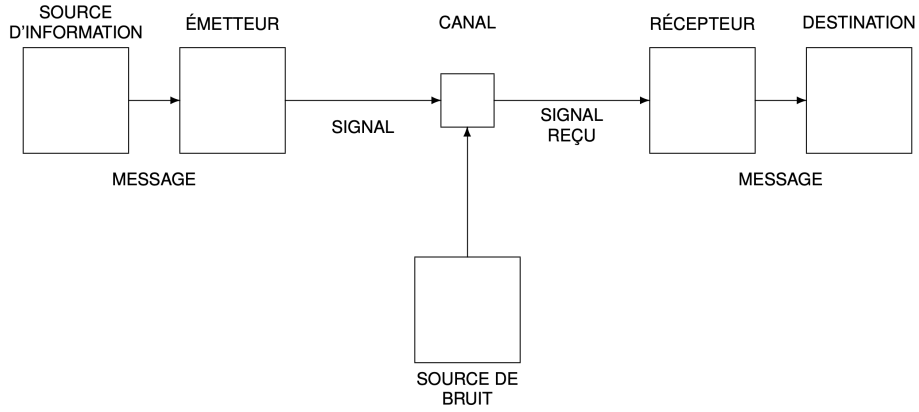
symboles émis par la source et les probabilités de transition indiquent la probabilité que le canal transmette correctement un symbole de la source au récepteur.



Dans cet exemple simplifié les flèches indiquent les probabilités de transition. Une chaîne de Markov est un processus aléatoire $(X_n)_{n\in N}$ dont les transitions sont données par une matrice stochastique $P(X_n, X_{n+1})$. Une matrice stochastique est une matrice carrée dont la somme des lignes fait 1 et dont chaque élément est un réel positif.

2.6. **Théorème de Shannon.**

Claude Shannon a modélisé un système de communication de la manière suivant :



Il se compose d'une source d'information qui génère un message. Ce message est ensuite transformé en signal par l'émetteur avant d'être transmis dans le canal. Le récepteur effectue l'opération inverse de l'émetteur, il transforme le signal en un message transmis au récepteur. Comme je l'ai mentionné précèdemment ce canal peut être contaminé par du bruit, dans ce cas le récepteur va faire une estimation du message avant de le délivrer à la destination.

Le théorème du codage de canal de Shannon énonce que si nous avons une source ayant une entropie H (bits par symbole) et un canal ayant une capacité C (bits par seconde) alors il est possible de coder la sortie de la source de manière à transmettre à un taux moyen de C/H -ϵ symboles par seconde sur le canal, où ϵ est arbitrairement petit. Il n'est pas possible de transmettre à un taux moyen supérieur à C/H. Cela signifie qu'il est

possible de transmettre des données numériques sur un canal bruité avec un faible taux d'erreur si le débit est inférieur à une certaine limite propre au canal. [3]

2.7. **Implémentation du jeu WORD GAME en utilisant la notion d'entropie.**

Rappelons les règles du jeu. Le but est de trouver un mot de 5 lettres parmi une liste de mots (de taille 12971 ici). À chaque proposition de mot, un résultat de 5 chiffres est retourné, avec une correspondance sur les lettres du mot :

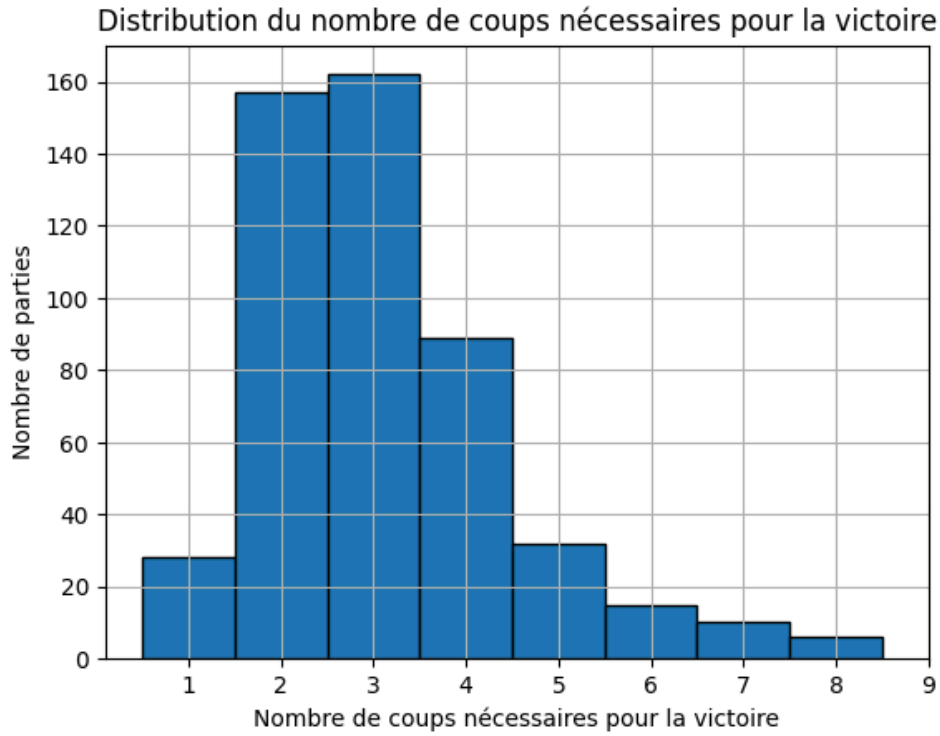
- 0 si la lettre n'est pas dans le mot
- 1 si la lettre est dans le mot mais pas à la bonne place
- 2 si la lettre est bien placée dans le mot

Pour résoudre efficacement ce jeu, nous allons chercher à maximiser l'information que l'on va recevoir de nos propositions de mots. Pour cela, nous allons utiliser l'entropie des mots. Il faut d'abord connaître, pour chaque mot, lequel va nous donner le plus d'information i.e celui qui a l'entropie la plus grande.
Pour définir le meilleur mot et maximiser l'information il faut pour chaque mot :

- établir, pour chaque pattern ([0,0,0,0,0],[0,0,0,0,1] , ... , [2,2,2,2,2]) combien de mots peuvent correspondre
- puis faire la moyenne du nombre de mots retourné en fonction des patterns pour obtenir l'entropie du mot

Après chaque proposition de mot, on élimine les mots qui ne peuvent pas correspondre et on propose un nouveau mot, celui avec la plus grande entropie, parmi les mots restants.

Voici, pour 500 parties, la répartition du nombre de coups nécéssaires pour trouver le mot. On remarque qu'il arrive que certaines parties se gagnent en plus de 5 coups. Cela survient lorsque le mot à deviner ressemble à une lettre près à d'autres mots. Dans ce cas, l'algorithme n'a d'autre choix que de tester les lettres au hasard et peut donc jouer jusqu'a 8 coups.



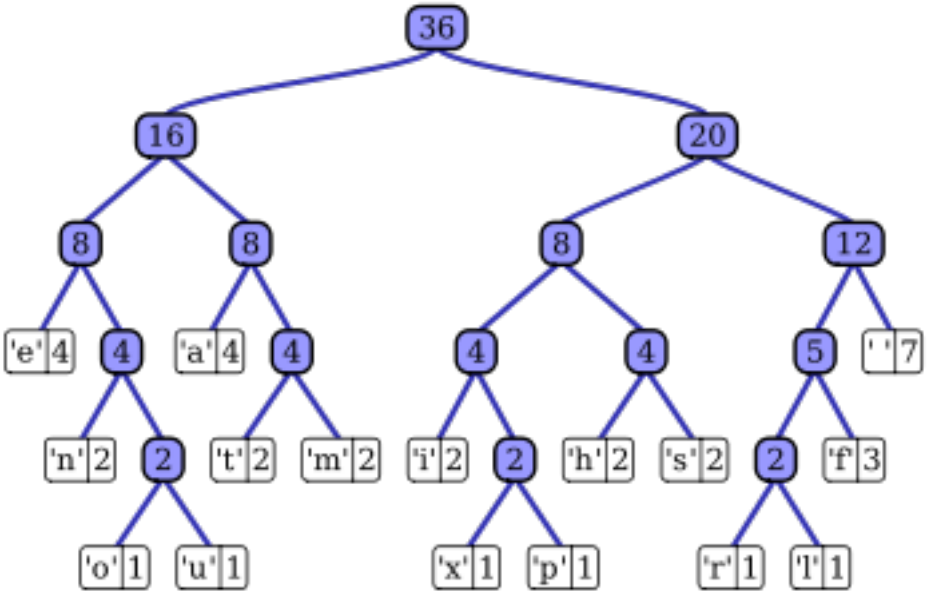
3. Application de la théorie de l'information

La théorie de l'information a de nombreuses utilités dans divers domaines informatiques comme la compression de données, les codes correcteurs ou encore la cryptographie. Pour chacun de ces exemples, nous allons expliquer comment la théorie de Shannon a permis de contribuer à ces domaines puis nous les illustrerons par des exemples.

3.1. **Compression de données - Huffman.**

Nous avons choisi le codage de Huffman comme application pratique de la théorie de l'information et notamment de la notion de redondance. Le codage de Huffman est un algorithme de compression de données sans perte. Il se base sur un arbre binaire composé de nœuds, dans ces nœuds se trouvent des caractères avec leur poids qui correspond à leur nombre d'occurences. On associe ensuite 2 nœuds avec les poids les plus faibles pour obtenir un noeud parent dont son poids est la somme des 2 poids de ses fils.

Nous avons généré l'arbre suivant à partir de la phrase "this is an example of a huffman tree". Si on additionne le poids de tous les nœuds on obtient 135 bits au lieu de 288 bits (36 caractères * 8 bits).



Vous retrouverez en annexe un code python qui illustre cette exemple, nous créons dans un premier temps un arbre de Huffman et nous compressons le message "this is an example of a huffman tree" ce qui nous donne "01101100101001011111010010111110000101110000011010001111011000111100011101001110111110011111000100011011101011110000101110110101110000000" (135 bits). Lorsque nous déchiffrons ce message binaire nous obtenons bien le message d'origine.

3.2. **Code correcteur d'erreur - Hamming.**

Le code de Hamming est un code détecteur d'erreurs. Dans ce code, des bits de redondance sont ajoutés dans le message original pour permettre la détection et la correction d'erreurs. Le code de Hamming permet la détection et la correction d'une erreur.

Pour encoder un message dans le code de Hamming, il faut suivre ces 3 étapes :

- Calcul du nombre de bits redondants
- Vérification de la position des bits redondants
- Calcul de la valeur des bits redondants

3.2.1. *Calcul du nombre de bits de redondance.*

Soit m le nombre de bits de données d'information et k le nombre de bits de parité. Le message contient donc $n = m+k$ bits. Les k bits de contrôle doivent pouvoir détecter les $n+1$ possibilité d'erreurs, il faut donc k tel que $2^k \geq n + 1$.
Voici un tableau permettant de déterminer k en fonction de n

m	0	0	1	1	2	3	4	4	5	6	7	8
k	1	2	2	3	3	3	4	4	4	4	4	4
n	1	2	3	4	5	6	7	8	9	10	11	12

Il faut ensuite numéroter les bits de droite à gauche à partir de 1. Les bits de contrôle sont placés au niveau des puissances de 2 (en position 1 (2^0), 2 (2^1), 4 (2^2) etc ...). Ces bits de contrôle effectuent un contrôle de parité sur les bits de données.
Déterminons ensuite quels bits de données contrôlent les bits de contrôle.
Supposons que nous avons un message contenant 4 bits d'informations (*m=4*), on peut construire un code de Hamming sur 7 bits (*n=7*). On ajoute donc 3 bits de contrôle (*k=3*)
Après avoir placé les bits de contrôle en position 2^k (1,2,4) on obtient ce code :

7	6	5	4	3	2	1
m4	m3	m2	k3	m1	k2	k1

Les bits d'information se trouvent en position 3,5,6 et 7. Le bit d'information en position 3 est contrôlé par les bits en position 1 et 2 car $1 + 2 = 3$. Le bit en position 5 est contrôlé par les bits en position 4 et 1($4 + 1 = 5$) et ainsi de suite. Pour déterminer la parité des bits de contrôle, il faut regarder la parité des bits qu'ils contrôlent. Par exemple k1 contrôle les bits 3,5,7. Si le nombre de bits à 1 sur les bits 3,5,7 est impair, le bit k1 sera 1. Si le nombre de bits à 1 est pair, le bit k1 sera 0. Une fois la parité des bits de contrôle effectuée, on peut envoyer le message.
Pour contrôle un code de Hamming reçu, il faut d'abord compter le nombre de bits transmis. Par exemple, si l'on reçoit le message : 1011100 Nous avons $n = 7$ donc $k = 3$ et $m = 4$

7	6	5	4	3	2	1
m4	m3	m2	k3	m1	k2	k1
1	0	1	1	1	0	0

Ici k1 = 0. Or k1 contrôle les bits (1,3,5,7) et le nombre de 1 est impair (3). Donc le bit de contrôle k1 est faux. k2 = 0 et k2 contrôle les bits (2,3,6,7) donc le bit k2 est bon. Par le même principe, le bit k3 est faux. Il est donc maintenant possible de trouver l'adresse de l'erreur qui est en position k1k2k3 c'est à dire 101 donc ici le bit 5. Ce bit en position 5, qui vaut 1 est donc faux. En corrigeant ce bit et en enlevant les bits de parité, on retrouve le message 1001.

3.3. **Cryptographie.**

3.3.1. *Introduction à la cryptographie.* Dans une aire où le numérique est partout, il est indispensable de protéger nos informations, nos échanges ... Ici, nous allons nous intéresser à la cryptographie qui est une discipline informatique visant à protéger des messages d'attaquants malveillants. En cryptographie, il y a plusieurs concepts clés sur lesquels jouer:

- **La confidentialité** qui est le fait d'empêcher tout accès non autorisé à des informations sensibles
- **L'authenticité** s'assure que la personne qui nous envoie le message est bien celle à laquelle on s'attend et inversement elle s'assure qu'on envoie notre message à la bonne personne
- **L'intégrité** consiste à protéger le message d'éventuelles modifications non désirées ainsi que de garder de la cohérence

Dans un cryptosystème nous allons rechercher trois qualités qui sont [4] :

- la confusion, aucune propriété statistique ne peut être déduite du message

chiffré.

- la diffusion, toute modification du message en clair se traduit par une

modification complète du message chiffré.

- la robustesse de la clé, difficulté à trouver K ou à énumérer tous les k possibles.

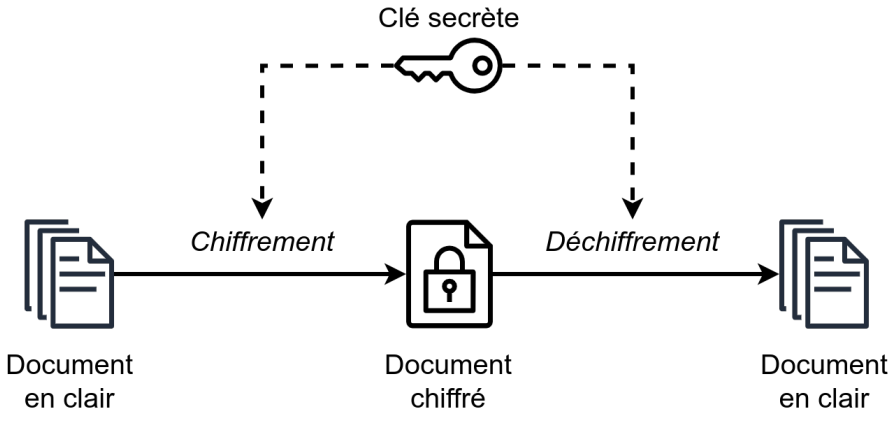
De nos jours, la cryptographie se base sur 2 systèmes au fonctionnement différent, la cryptographie à clé secrète et à clé publique. Dans les deux cas, il faut générer des clés aléatoires qui se basent sur des calculs mathématiques complexes comme l'arithmétique modulaire (utilisée par RSA), des fonctions de hachages, des courbes elliptiques, etc.

3.3.2. *Cryptographie à clé secrète.*

La cryptographie à clé secrète aussi appelée cryptographie symétrique chiffre et déchiffre le message à l'aide d'une même clé.

Le processus se déroule de la façon suivante :

- Les 2 parties se mettent d'accord sur la longueur de la clé puis la génèrent aléatoirement. Une fois ceci fait, il est crucial de garantir que cette clé reste confidentielle entre les parties autorisées sinon il y est simple pour un tiers non autorisé de déchiffrer le message.



Parmi les autres inconvénients, en plus de devoir partager la clé, il y a le nombre de clés à échanger entre plusieurs personnes.

Nombre de personnes	Nombre de clés
2	1
5	10
100	4450
1000	499500
n	$\frac{n(n-1)}{2}$

Parmi les algorithmes à clé secrète connus et utilisés, il y a le chiffrement de Vigenere, affine , DES, AES pour n'en citer que quelques-uns. Nous allons traiter le cas du chiffrement affine pour comprendre son fonctionnement.

3.3.2.1. *Chiffrement affine.*
Le chiffrement affine est un chiffrement par substitution mono-alphabétique c'est à dire que chaque lettre est remplacée par une unique lettre. Pour cela, il faut utiliser une fonction de chiffrage affine $y = ax + b$
Les lettres *a* et *b* sont constituent la clé privée, l'unique condition porte sur *a* qui doit être premier avec 26 pour pouvoir coder correctement chaque lettre. La première étape consiste à associer à chaque lettre à un nombre (A = 0, B = 1 ... Z = 25) Il faut ensuite calculer la valeur de y pour chaque lettre. La valeur calculée doit être *modulo 26* pour que le résultat reste un nombre entre 0 et 25. Une fois la valeur de chaque y calculée, on re transforme le nombre obtenu en lettre avec la même correspondance (1 = A, 2 = B)
Par exemple pour le mot *information*

- La transformation de chaque lettre en nombre donne [8, 13,5,14,19,12,0,19,8,14,13]
- Avec la clé a=3 et b=5 et la fonction affine $y = ax + b \bmod 26$ on obtient [3,18,20,21,4,15,5,10,3,21,18]
- En convertissant ces nombres en lettres, les messages cryptés obtenus sont : *dsuvepfkdvs*

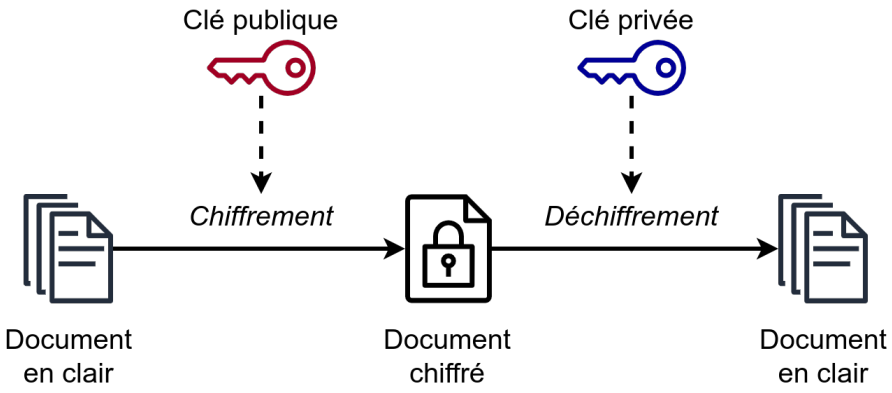
Pour déchiffrer le message obtenu, toujours à l'aide de la clé (*a,b*), il faut procéder de la même façon avec la relation inverse $y = a^{-1}x - b \equiv 1$ mod 26. Il faut donc calculer l'inverse de *a* c'est à dire a^{-1} tel que $aa^{-1} \equiv 1 \bmod 26$. Ici l'inverse de *a* est 9 ($6 * 9 = 1 \bmod 26$). En appliquant, pour chaque nombre du message la relation $y = (9 * x - 5) \bmod 26$ et en convertissant les nombres obtenus en lettre, on retrouve le message initial.
Cette façon de crypter l'information, de par sa simplicité, est extrêmement vulnérable. Il est très simple de craquer le code et de retrouver la clé publique par *brute-force*. Il suffit de tester toutes les valeurs de *a* et de *b* jusqu'a l'obtention du message correct.

3.3.3. *Cryptographie à clé publique.*

Le principe de la cryptographie à clé publique s. e base sur plusieurs clés, une clé privée et une clé publique. La clé publique est disponible de tous et toute personne qui la connaît peut envoyer un message chiffré au propriétaire de la clé et une clé publique. La clé privée quant à elle est confidentielle mais liée à la clé publique correspondante via un lien mathématique (les fonctions à sens unique), elle sert à déchiffrer les messages chiffrés avec la clé publique associée.

Une fonction à sens unique est une fonction qui est facile à calculer dans un sens mais difficile à résoudre dans l'autre sens. Il est facile de calculer $f(x)$ en ayant x mais très dur de trouver x en ayant $f(x)$. On retrouve par exemple la factorisation d'un entier ou le logarithme discret.

Prenons l'exemple de Bob qui souhaite envoyer un message à Alice, Bob va récupérer la clé publique de celle-ci puis chiffrer son message avec. Il envoie son message à Alice qui va déchiffrer le message avec sa clé privée.



Parmi les algorithmes à clé publique connus et utilisés il y a le protocole de Diffie-Hellman ou encore RSA.

3.3.3.1. *Protocole d'échange de clé de Diffie-Hellman.*
Ce protocole, proposé en 1976, est un protocole d'échange de clés sécurisé. Il permet, pour 2 personnes qui veulent s'échanger des informations avec une clé sur un canal non sécurisé, de générer une clé que seules ces 2 personnes peuvent connaître. Ce protocole se base sur l'arithmétique modulaire et le principe suivant :

- Avec des entiers p,a,x et p premier et $1 \leq a \leq p-1$, il est facile de calculer $y = a^x \bmod p$
- Si on connaît la valeur y,a et p, il est difficile, si p est grand, de retrouver la valeur de x

Prenons le cas d'Alice et Bob qui souhaitent s'échanger une clé secrète via le protocole Diffie-Hellman.

- Alice et Bob choisissent ensemble un grand nombre p premier et un entier a tel que $1 \leq a =< p-1$

- Alice choisit de son côté un entier *x1* et Bob fait de même avec *x2*
- Alice calcule $y_1 = a^{x1} \bmod p$ et Bob calcule $y_2 = a^{x2} \bmod p$
- Alice envoie y_1 à Bob et Bob envoie y_2 à Alice
- Alice calcule $y_2^{x1} = \left(a^{x2}\right)^{x1} = a^{x1x2} \bmod p$ et Bob calcule $y_1^{x2} = \left(a^{x1}\right)^{x2} = a^{x1x2} \bmod p$

Le résultat de ces 2 calculs est *K* et correspond à la clé secrète.

À sa création, ce protocole a révolutionné l'histoire de la cryptographie. Mais son inconvénient est qu'il nécessite que les 2 personnes fassent les actions en même temps. Il reste néanmoins utilisé dans la technologie Bluetooth pour l'appariement de deux appareils.

4. Conclusion

Nous venons de voir les différentes applications du théorème de Shannon. Sa formalisation du concept de l'information en s'appuyant sur une base mathématique a transformé la façon de manipuler l'information. Claude E. Shannon a introduit le concept de *bit* comme une unité de mesure quantifiable pour l'information. Cela a permis d'évaluer précisément la quantité d'informations transmises dans un message.
En empruntant la notion d'entropie à la thermodynamique et en l'adaptant pour caractériser l'incertitude d'une information, Shannon a réussi à mesurer la redondance dans un message, ainsi que le bruit dans un canal de communication. Cela a ensuite permis d'élaborer des algorithmes plus efficaces pour corriger les erreurs (`code de Hamming`) ou compresser des données avec par exemple la `compression d'Huffman`.
La théorie de l'information de Shannon a contribué à faire avancer le domaine de la cryptographie. Grâce à une meilleure compréhension de la façon dont l'information est quantifiée, les méthodes de chiffrement ont pu être améliorées.
Ses travaux continuent à ce jour, d'être au cœur du développement de nouvelles technologies de communication.

References

1. Cours: Maria-João Rendas, `https://webusers.i3s.unice.fr/~rendas/SICOM/Lectures1.pdf`
2. Passe-Science: La théorie de l'information de Claude Shannon - Passe-science. (2021)
3. C. E. Shannon: A Mathematical Theory of Communication, (1948)
4. Pierre Ramet: Cryptographie asymétrique, `https://ramet.gitlab.io/r3.09-crypto/asymetrique.pdf`
5. Olivier Rioul: Qu'est-ce que la théorie de l'information ?, `https://culturemath.ens.fr/thematiques/probabilites/qu-est-ce-que-la-theorie-de-l-information`
6. Olivier Rioul: Shannon et la théorie de l'information, `http://www.bibnum.education.fr/sites/default/files/174-shannon-analyse.pdf`
7. Leroux Joel: Deuxième théorème de Shannon, `https://users.polytech.unice.fr/~leroux/transmission/node42.html`

8. Raphael Kassel: Le codage de Huffman Description et mode d'emploi, https://datascientest.com/codage-de-huffman-tout-savoir

9. David Blackwell: Entropie, https://www.britannica.com/biography/David-Blackwell

10. Damien Ecrohart: Confidentialité, intégrité et disponibilité - Application dans le monde réel, https://blog.netwrix.fr/2020/06/23/confidentialite-integrite-et-disponibilite-application-dans-le-monde-reel/

11. Le chiffre affine, https://www.bibmath.net/crypto/index.php?action=affiche&quoi=lexique/affine

12. J.CHIMBAULT: Encodage de l'information, http://workig.free.fr/ch07.html

13. 3Blue1Brown: Solving Wordle using information theory. (2022)

Département informatique, Université de Bordeaux, Gradignan, 33170