MASTER I CIC – UTBM
2020 – 2021

FONDEMENTS THEORIQUES DE L'INFORMATIQUE

LES MATHEMATIQUES ET LA COMPRESSION DES DONNEES

PRESENTE PAR :

ATTIOGBE Koffi James
FIOKLOU Amé Ludovic
MAVANGA - VUNGA Benel

CHARGE DU COURS :

Dr. Edarh BOSSOU

RESUME

La compression est une réduction du nombre de bits nécessaires pour représenter les données. Compresser les données permet d'optimiser la capacité de stockage et la vitesse de transfert des fichiers ; ce pourquoi, la compression est actuellement très sollicitée par le domaine de l'informatique, fin de gagner en efficacité et en rapidité.

Plusieurs méthodes de compression ont vu le jour et ceci, en partie à cause des différents types de données qui existent : audio, vidéo, image, texte, etc.

Dans cet exposé, nous aurons à passer en revue les bases de la théorie de l'information, de l'algorithme de compression, de nombre premier ainsi que de l'analyse informatique – mais du point de vue mathématique. Le tout dans l'optique de présenter un algorithme qui se repose sur un modèle mathématique (Les nombres premiers). Plus concrètement, dans notre cas, d'arriver à trouver une fonction mathématique qui attribue à chaque Séquence "S", des caractères à compresser, un entier E.

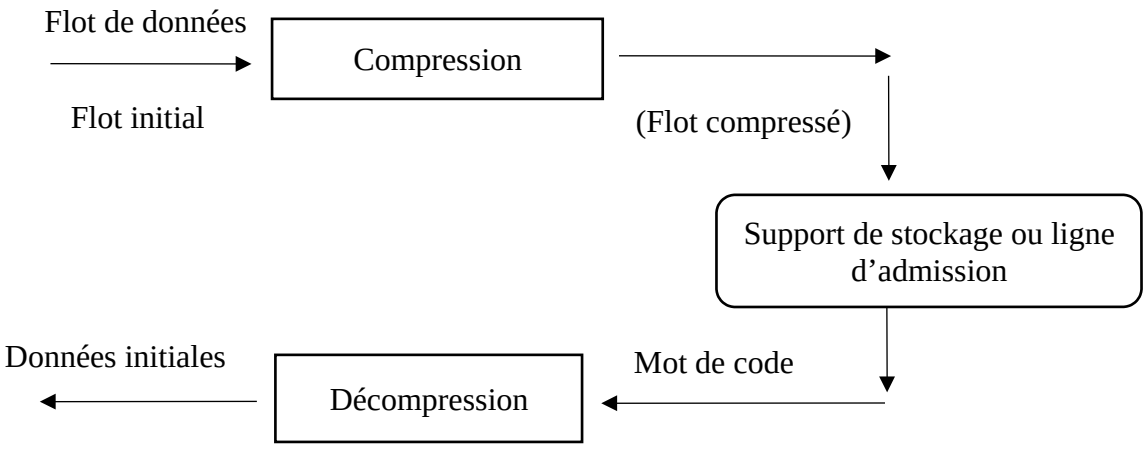
TABLE DE MATIERE

INTRODUCTION

La quantité d'information échangée entre usagers, ayant considérablement pris augmenté, nécessite de supports de stockage alliant puissance et performance dans l'utilisation de stockage et de la bande passante limitée, étant donnée la forte utilisation. C'est là qu'intervint la compression, en tant que solution idéale pour prendre en charge ces problèmes.

La compression consiste à réduire la taille des données, permettant à un processeur d'atteindre une vitesse optimale de traitement de données, et de la transmission de données. De ce fait, compresser est le fait de réduire la taille physique d'un bloc d'information. Elle repose sur la recherche de ressemblances au niveau de la forme et du motif. Ainsi, au lieu d'enregistrer une image complète, ou même une vidéo/audio, il suffit d'en enregistrer la description en utilisant des tables ou des algorithmes mathématiques, faisant résulter une optimisation de traitement du fait qu'il devient possible de placer plus d'informations dans un même stockage et de réduire par la même occasion, le temps de transfert des données sur un réseau.

Voici ci-dessous, le schéma de compression/décompression



0. THEORIE DE L'INFORMATION

La théorie de l'information s'est avérée fondamentale avec l'élaboration de la théorie mathématique par SHANON et WEAVER en définissant les concepts de quantité d'information, d'entropie[1] et de redondance que nous allons décrire :

0.1. QUANTITE DE L'INFORMATION

La définition mathématique de la quantité d'information sur une théorie probabiliste : la quantité d'information contenue dans un flot de données est liée à sa probabilité d'apparition.

Soient $S=X_1,X_2,X_3,\ldots,X_k$, une source d'information de K symboles (alphabet) et $P(X_i)$la probabilité d'apparition de X_i .

La quantité d'information (I_i) contenue dans un flot de données est une fonction de l'inverse de la probabilité d'apparition de ce symbole. Elle est donnée par la relation suivante :
$I_i=\log_2(1/P(X_i))=-\log_2(P(X_i))$. L'unité de la quantité d'information est le **Shannon**.

0.2. ENTROPIE

En général, l'entropie mesure le degré de désordre dans un système donné. En théorie d'information, l'entropie mesure le degré du hasard dans un flot de données (fichier). Un fichier à faible taux d'entropie est considéré « très compactable » car ses éléments sont tous prévisibles. Par exemple, la quantité d'information contenue dans un fichier dont tous les éléments sont nuls est nulle. Dans ce cas, le hasard est inexistant et donc l'entropie minimale. Inversement, un fichier *zippé*, donc compacté, contient une suite aléatoire d'octets sans aucune relation les uns avec les autres. La quantité d'information y est maximale car, à priori, toutes les redondances ont été supprimées. Dans ce cas, le hasard est maximal et donc l'entropie maximale.

L'entropie H d'un fichier est l'information moyenne contenue par chaque symbole, et est donnée par la relation :

$H=\sum I_i P(X_i)=-\sum P(x_i)\log_2(P(X_i))$; où

- $0\le H\le\log_2(n)$
- $H=0$ lorsque $P(X_i)=1$
- $H=\log_2(n)$ lorsque $P(X_i)=1/n, i=1..n$

1 Entropie : L'**entropie de Shannon**, due à Claude Shannon, est une fonction mathématique qui, intuitivement, correspond à la quantité d'information contenue ou délivrée par une source d'information. Cette source peut être un texte écrit dans une langue donnée, un signal électrique ou encore un fichier informatique quelconque (collection d'octets).

0.3. REDONDANCE

La compression de données se fonde sur la détection et l'élimination des redondances dans un flot de données, sachant que l'information en informatique (texte, audio, image, etc.) est représentée par une combinaison de 256 octets, appelée alphabet. Ceci implique que chaque flot de données de taille supérieure à celle de l'alphabet (nombre de symboles) présente nécessairement une redondance.

Exemple : Soient « A » l'alphabet et « F » le flot de données.

Si $A=\{0,1\}$, sa taille est de 2 symboles. Un flot de données $F=1001\ldots10$, présente une redondance de symboles 0 et 1 dès que la taille de F dépasse 2. De ce fait, les flots de données de grande taille (plusieurs Mégaoctets) ne sont que le résultat de la redondance des symboles de l'alphabet qui est un ensemble borné.

1. ALGORITHME DE COMPRESSION

Il existe un très grand nombre de type d'algorithme de compression, mais dans notre cas, nous en retenons deux :

- Les algorithmes irréversibles (*lossy*) qui adoptent une compression de données en permettant une légère perte d'information. Ils s'appliquent aux des flots de données comme les images et le son.
- Les algorithmes réversibles (*lossless*) qui sont applicables aux flots de données qui ne tolèrent aucune perte d'information tels que les textes et les programmes.

Ces algorithmes sont fondés sur des techniques de codages.

Le codage est une fonction qui fait correspondre à chaque symbole (ou groupe de symboles) du flot de données à compresser, des symboles compacts (codes).

Pour avoir un bon codage des données, le code doit vérifier les propriétés suivantes :

- Tous les mots du code peuvent être distingués.
- Le décodage ne doit pas donner lieu à aucune ambiguïté.
- Aucun mot du code n'est un sous mot initial d'un autre.

Dans notre cas, nous avons identifié un moyen de codage qui fait appel à la théorie des nombres premiers et qui répond aux différentes conditions citées ci-dessus. Nous le décrivons dans la section suivante.

2. LES NOMBRES PREMIERS

La reconnaissance des nombres premiers et des nombres composés avec leur décomposition en facteurs premiers est connue pour être un champ important et utile en arithmétique. Le théorème fondamental de l'arithmétique[2] stipule que :

- Tout nombre entier naturel est décomposable de façon unique en produit de ses diviseurs premiers ;
- Les facteurs premiers sont des nombres premiers ;
- Tous les nombres premiers sont impairs, sauf 2,
- Tout nombre premier impair est de la forme 4k+1 ou 4k + 3.

Il existe par ailleurs trois propriétés fondamentales[3] qui décrètent que :

- Aucune formule algébrique n'existe pour représenter un nombre premier ;
- Une infinité de nombres premiers existe ;
- La factorisation d'un nombre en facteurs premiers est unique.

3. ANALYSE

À partir du théorème fondamental de l'arithmétique, découle l'idée de compresser des données en utilisant les propriétés des nombres premiers. En fait, l'unicité de la décomposition en facteurs premiers d'un entier favorise la réversibilité de la fonction de compression (décompression sans ambiguïté).

3.1. CODAGE

La fonction de codage utilisée, attribue à chaque octet du flot de données, un nombre premier de manière statique (indépendamment du contenu de ce flot) ou dynamique en attribuant le plus petit nombre à l'octet le plus fréquent. Exemple : Le tableau 1 présente un codage statique en attribuant un nombre premier différent pour chaque octet du flot.

Octet	**Code**
00000000	2
00000001	3
00000010	5
00000011	7
00000100	9
…..	…
C_i	X_i
11111111	X_n

2 Arnaudiès, Fraysse, 1977
3 Lehning, Jakubowicz, 1982

3.2. FONCTION DE COMPRESSION/DECOMPRESSION

La fonction de compression est fondée sur la composition des nombres premiers correspondants à chaque octet qui apparaissent dans le flot de données. Cette fonction transforme chaque chaîne de caractères du flot de données en un seul nombre entier de la manière suivante :

Si $C = « C_1 C_2 C_3 \dots C_n »$ alors,

$$F(C) = F(C_1, C_2, C_3, \dots C_n) = P(C_1) * P(C_2) * P(C_3) * \dots P(C_i) \dots * P(C_n) = N \dots (1)$$

Où :

- C_i est l'octet à la position i de la chaîne C ;
- $P(C_i)$ est le nombre premier qui correspond à l'octet C_i;
- N est le nombre entier résultat de la compression de la chaîne C.

La fonction de décompression est la décomposition du nombre N en facteurs premiers qui génère l'ensemble des caractères de la chaîne initiale (unicité de la décomposition). Cependant, un problème réside du fait que nous ne pouvons pas décider de la position du caractère dans la chaîne initiale.

Illustrons cela avec l'exemple de la chaîne « ab » et de la chaîne « ba ». Dans ce cas,

$F(a,b) = F(b,a) = P(a) * P(b)$. Ce qui ne permet pas de distinguer les deux chaînes « ab » et « ba ».

Pour pallier alors ce problème et permettre à la fonction de décompression de donner (exactement) la chaîne initiale, nous avons introduit le paramètre « position » dans la formule *(1)*. La fonction de compression devient :

$$F(C) = F(C_1, C_2, C_3, \dots C_n) = (P(C_1))1 * (P(C_2))2 * \dots * (P(C_i))i * \dots * (P(C_n))n = N \dots (2)$$

Où i est la position du caractère dans la chaîne.

L'exemple de la chaîne « ab » et de la chaîne « ba » est ainsi résolu. Pour l'illustrer, supposons que $P(A) = 2 \; et \; P(B) = 3$. Dans ce cas, le processus de compression produit deux valeurs différentes pour $F(a,b)$ et $F(b,a)$. En effet, $F(a,b) = 2^1 * 3^2 = 18$ et $F(b,a) = 3^1 * 2^2 = 12$.

Le processus de décompression permet de retrouver exactement la chaîne initiale :

$$F^{-1}(18) = 2 * 3 * 3 = 2^1 * 3^2 = « ab »$$
$$F^{-1}(12) = 2 * 2 * 3 = 3^1 * 2^2 = « ba »$$

Une telle fonction est caractérisée par les propriétés de l'associativité et la non-commutativité. Elle est également bijective (tout entier du flot compressé correspond à une et une seule chaîne du flot de données initial). La compression est ainsi garantie sans ambiguïté et sans perte. Toutefois, pour prendre réellement en charge l'application de la fonction de compression $F()$ avec plus d'efficacité, les contraintes suivantes doivent être respectées.

3.2.1. *Contrôle du débordement*

La fonction $F(S)$ ne fait pas de restriction sur la taille de la chaîne S. La taille du résultat (N) qui ne doit pas dépasser 216-1 (un entier sur 16 bits) dépend de la taille de S. Il est donc important d'effectuer un contrôle sur le débordement. Ce contrôle est simplifié par la propriété de l'associativité de la fonction $F()$. La composition des nombres premiers est interrompue dès l'apparition d'une situation de débordement.

Exemple : soient $S=«GFUY\ldots..X»$ et $Max(E)$: le plus grand nombre entier non signé.

$$F(G,F,U,Y,\ldots,X)=P(G)^1*P(F)^2*P(U)^3*P(Y)^4*\ldots*P(X)^K\ldots(3)$$

Supposons que l'expression

$P(G)^1*P(F)^2*P(U)^3<Max(E)<P(G)^1*P(F)^2*P(U)^3*P(Y)^4$.

Le résultat partiel est alors sauvegardé dans le flot de sortie et une nouvelle compression de $S'=«YP\ldots..X»$ est initiée. $F(G,F,U,Y,\ldots,X)$ devient $F(G,F,U)F(Y,\ldots.X)$.

3.2.2. *Non ambiguïté de la position des caractères*

Pour illustrer la pertinence de cette contrainte, prenons l'exemple des chaînes S = « abba » et S' = «baab».

$F(S)=F(a,b,b,a)=P(a)^1*P(b)^2*P(b)^3*P(a)^4=P(a)^5*P(b)^5$
$F(S')=F(b,a,a,b)=P(b)^1*P(a)^2*P(a)^3*P(b)^4=P(b)^5*P(a)^5$
$F(S)$ et$F(S')$sont identiques, créant ainsi une ambiguïté.

Dans ce cas, la fonction de décompression ne peut pas conclure sur la position des caractères a et b. Pour résoudre cela, il faut éviter de compresser la totalité de la chaîne qui présente un caractère redondant et procéder à une décomposition de la chaîne. Ainsi, F(abba) peut être traité comme F(ab) F(ba) comme illustré ci-dessous.

$F(S)=F(a,b)F(b,a)=[P(a)^1*P(b)^2][P(b)^1*P(a)^0]=[N1][N2]$
$F(S')=F(b,a)F(a,b)=[P(b)^1*P(a)^2][P(a)^1*P(b)^2]=[N'1][N'2]$

Il en résulte que F(S) et F(S') ont des valeurs différentes permettant la résolution de l'ambiguïté initiale.

3.3. AVANTAGE DE LA METHODE

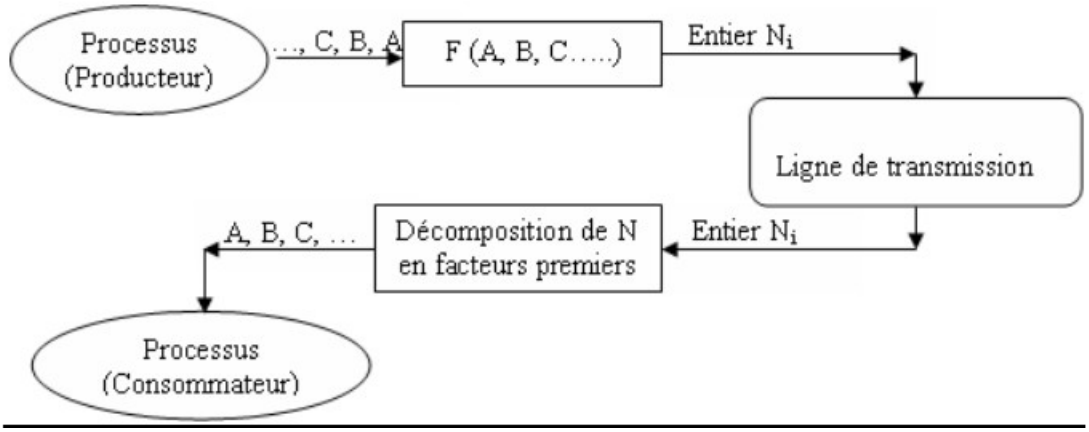
Nous pouvons citer principalement deux avantages que nous décrivons ci-dessous.

3.3.1. *Une simplicité de mise en œuvre*

Outre la simplicité du modèle mathématique utilisé, le non-recours à un dictionnaire réduit considérablement la taille du programme.

3.3.2. *Une adéquation à la compression temps réel*

Le principe utilisé offre la possibilité de compresser un flot de données temps réel puisque la fonction de composition est appliquée au fur et à mesure que les octets « arrivent » sans attendre la constitution de la totalité du fichier. Dans cette configuration, la quantité de données transmise sur le réseau sera réduite simulant un débit virtuel plus grand (envoi des chaînes de caractères sous forme d'entiers de petite taille) et par conséquent, des délais réduits, donnant une meilleure interactivité aux applications temps réel.



4. CONCLUSION

Dans cet exposé, nous avons proposé une méthode pour la compression de données, une méthode qui est fondée sur la théorie des nombres premiers, proposant une façon de faire sans que produire de redondances. Améliorant ainsi le taux de compression en utilisant des techniques de compression qui les éliminent et en usant d'un algorithme qui offre la faculté de compresser des flots de données en ayant une entropie maximale.

REFERENCES

Sites consultés :

https://www.lesechos.fr/2016/04/claude-shannon-pere-de-lage-numerique-1110851

https://hal.archives-ouvertes.fr/hal-01343890v3

https://hal.archives-ouvertes.fr/hal-01343890v3

https://fr.wikipedia.org/wiki/Entropie_de_Shannon#:~:text=L'entropie%20de%20Shannon%2C%20due,par%20une%20source%20d'information.&text=Plus%20le%20r%C3%A9cepteur%20re%C3%A7oit%20d,vis%20de%20ce%20message%20cro%C3%AEt.

Documentation :

Claude Shannon et la compression des données, par Gabriel Peyrer

Algorithme pour la compression de données, par Sadou Samir, Mohamed Ramdane, Mustapha Lalam, Rachid Ahmed Ouamer.

Mark Nelson, (1992). Compression de données (images, son, texte). Paris : Edition DUNOD.

Pascal Plume, (1993). Compression de données : méthodes, algorithmes, programmes détaillés. Paris: Edition EYROLLES.

S.Maadi, Y.Peneveyre, C. Lambercy, (2002). Compression de données avec pertes. Suisse : IICT [Institute for Information and Communication Technologies].

R. Benzid, (2005). Ondelettes et statistique d'ordre supérieur appliquées aux signaux uni et bidimensionnels. Doctorat d'état en Electronique. Université de BATNA, Algérie.

D. Salomo, (1997). Data Compression -The Complete Reference. Berlin: Edition Springer.