Algorithme pour la compression de données

Sadou Samir

Université Mouloud Mammeri (Tizi-Ouzou), Rue Fettala, Cne Tifra 06412, Sidi-Aich, Bejaïa, Algérie sad_samir2003@yahoo.fr

Mohamed Ramdane

Université Mouloud Mammeri (Tizi-Ouzou), Village Ait Amar, Cne Ait Bouaddou 15460, Tizi-Ouzou, Algérie ramdane.moh@caramail.com

Mustapha Lalam

Laboratoire de Recherche en Informatique (LARI), Université Mouloud Mammeri, BP 17RP 15000, Tizi-Ouzou, Algérie

Rachid Ahmed Ouamer

Laboratoire de Recherche en Informatique (LARI), Université Mouloud Mammeri, BP 17RP 15000, Tizi-Ouzou, Algérie

Résumé

La compression de données est actuellement très sollicitée par le domaine de l'informatique en quête de solutions logicielles rapides et peu chères permettant au matériel utilisé, qui dans certains cas atteint ses limites en termes de débit et de capacité mémoire, de gagner en efficacité et en rapidité. Plusieurs méthodes de compression ont vu le jour. Cette variété de méthodes est due à la diversité des types de données ciblés : images, audio, vidéo, texte, etc. Cet article propose un algorithme de compression qui peut résoudre aussi bien le problème de l'espace de stockage que celui du débit critique pour les applications temps réel. L'algorithme proposé repose sur un modèle mathématique (théorie des nombres premiers). Il s'agit dans ce cas est de trouver une fonction mathématique réversible qui attribue à chaque séquence « S » de caractères à compresser, un entier « E ».

Abstract

Nowadays, The data compression is very requested by the field of data processing to offer fast and less expensive software solutions which can give more effectiveness and speediness for the material used which, in certain case, reached its limits (Throughput, memory size). Several methods of compression were born since the discipline appeared; this variety of methods is due to the different types of data: images, audio, video, text, etc.

This paper proposes an algorithm of compression which can be used as well to solve the space of storage problem as to solve that of the limited throughput especially in real time applications.

This algorithm is based on a mathematical model (Prime numbers theory). So, the challenge is to find a mathematical reversible function which attributes to each sequence "S" of characters from the source file an integer "E" in the target file (compressed file).

Mots-clés

algorithme de compression, nombres premiers, compression sans perte

Keywords

compression algorithm, prime numbers, lossless data compression

1. Introduction

De nos jours, grâce aux réseaux et à Internet particulièrement, la quantité d'information échangée entre les usagers a considérablement augmenté. Ceci nécessite des supports de stockage alliant puissance et performance pour satisfaire la forte demande de l'espace de stockage, et garantir un minimum de trafic sur un réseau qui souffre de bande passante limitée. Une solution très courante est la compression. Elle consiste à réduire la taille des données en exploitant la puissance des processeurs plutôt qu'en augmentant les capacités de stockage et de transmission des données.
Fondamentalement, La compression (cf. figure 1) est l'action utilisée pour réduire la taille physique d'un bloc d'information. La compression de données sur ordinateur repose sur la recherche de ressemblances au niveau de la forme ou du motif. Ainsi, au lieu d'enregistrer l'image complète, il suffit d'en enregistrer la description en utilisant des tables ou des algorithmes mathématiques. Il en résulte une optimisation du fait qu'il devient possible de placer plus d'informations dans un même espace de stockage et de réduire ainsi le temps de transfert des données sur un réseau.

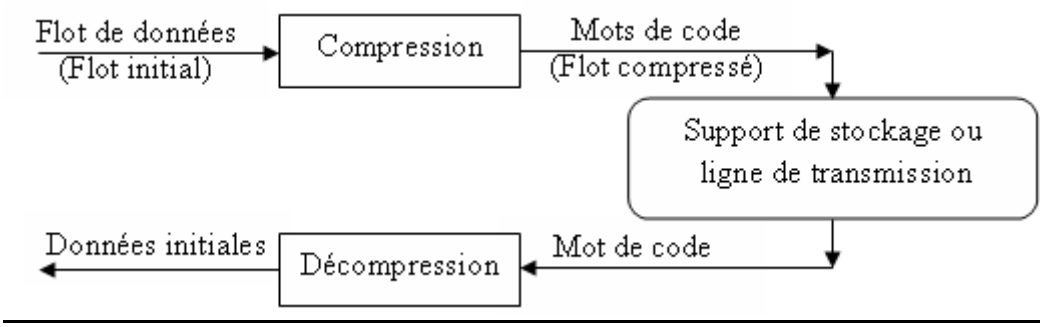

Figure 1. Schéma de compression/décompression

2. Théorie de l'information

La théorie de l'information (M. Nelson, 1992) (Pascal Plume, 1993) a pris sa dimension avec l'élaboration de la théorie mathématique par SHANON et WEAVER en définissant les concepts de quantité d'information, d'entropie et de redondance que nous décrivons ci-dessous.

2.1. Quantité d'information

La définition mathématique de la quantité d'information (S.Maadi, Y.Peneveyre, 2002) repose 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 : $I_i= \text{Log}_2 (1/P(X_i)) = -\text{Log}_2(P(X_i))$. L'unité de la quantité d'information est le Shannon.

2.2. Entropie

En général, l'entropie (Pascal Plume, 1993) (R**.** Benzid, 2005) (D. Salomo, 1997) 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 comptactable » 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[1] car, à priori, toutes les redondances ont été supprimées. Dans ce cas, le hasard est maximale et donc l'entropie maximale.

L'entropie H d'un fichier est l'information moyenne contenue par chaque symbole, elle est donnée par la relation $H=\sum I_i P(Xi) = - \sum P(xi)Log_2 (P(Xi))$; où

- $0 \leq H \leq Log_2 (n)$
- $H=0$ lorsque $P(Xi)=1$
- $H=Log_2(n)$ lorsque $P(Xi)=1/n$, $i=1..n$

2.3. La redondance

La compression de données se fonde sur la détection et l'élimination des redondances (M. Nelson, 1992) (S.Maadi, Y.Peneveyre, 2002) dans un flot de données. Sachant que l'information informatique (texte, audio, image, etc.) est logiquement représentée par une combinaison de 256 octets[2], 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…10, 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 Megaoctets) ne sont que le résultat de la redondance des symboles de l'alphabet qui est un ensemble borné.

3. Algorithmes de compression

L'éventail des algorithmes de compression de données est très large. Deux grands types d'algorithmes sont à distinguer :

- les algorithmes irréversibles (lossy) 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) 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 (D. Salomo, 1997) (Pascal Plume, 1993) 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.

4. Les nombres premiers

La reconnaissance des nombres premiers (Hardy, Wright, 1980) 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 (Arnaudiès, Fraysse, 1977) stipule que :

[1] Par définition puisqu'il s'agit d'un fichier déjà compressé

[2] On considère l'octet comme unité de base

- 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 (Lehning, Jakubowicz, 1982) 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
-

5. Notre contribution

À partir du théorème fondamental de l'arithmétique découle notre 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).

5.1 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	Xn

Tableau 1. Table de correspondance codes Ascii/Nombre premiers

5.2 Fonction de compression/décompression

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 donnée en un seul nombre entier de la manière suivante :
Si C = « $C_1C_2C_3…C_n$ » alors
$F(C) = F(C_1, C_2, C_3,…C_n) = P(C_1)*P(C_2)*P(C_3)*…P(C_i) …*P(C_n) = N$ ... (1)
Où :

- C_i est l'octet à la position i de la chaîne C.
- P(Ci) est le nombre premier qui correspond à l'octet Ci
- 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). Un problème réside

toutefois dans le fait que nous ne pouvons pas décider de la position du caractère dans la chaîne initiale comme l'illustre 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 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 :

$F(C)=F(C_1,C_2,C_3,\ldots C_n)=(P(C_1))^1*(P(C_2))^2*\ldots*(P(C_i))^i*\ldots*(P(C_n))^n=N$ ... (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, l'application de la fonction de compression F( ) avec plus d'efficacité, les contraintes suivantes doivent être respectées.
Contrainte1 : 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 2^{16-1} (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…..X» et *Max (E)*: le plus grand nombre entier non signé.

$F(G,F,U,Y,...,X) = P(G)^1 * P(F)^2 * P(U)^3 * P(Y)^4 * ... * P(X)^k$ … (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…..X» est initiée.
F(G,F,U,Y,...,X) devient F(G , F ,U) F(Y , ... X).
Contrainte 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.

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

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

5.3 Avantage de la méthode

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

- 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.
- 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 (cf. figure 2). Dans cette configuration, la quantité de données transmise sur le réseau (lignes de communication) 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 meilleur interactivité aux applications temps réel.
- **Favorable à l'entropie :** les méthodes de compression procèdent en général à l'élimination des redondances et par conséquent l'augmentation de l'entropie. La méthode que nous proposons est fondée sur une logique totalement différente, puisqu'au lieu d'éliminer les redondances, elle compresse des données qui ne présentent pas de redondance et donc une entropie maximale. Ce qui est en faveur d'une bonne compression puisqu'à chaque fois que l'entropie est grande, la compression est bonne selon la contrainte 2 de la méthode.

Les inconvénients de cette méthode résident dans ses contraintes qui peuvent faire l'objet de travaux futurs pour en améliorer le rendement.

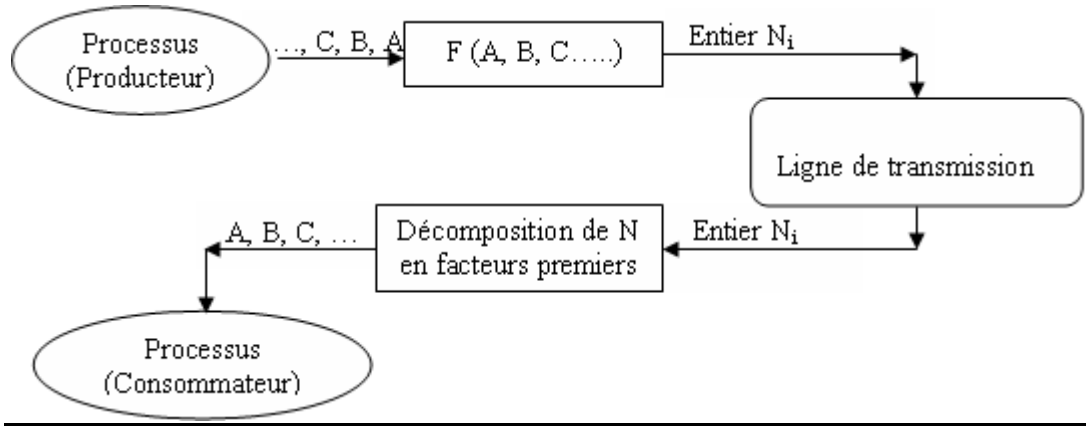

Figure 2. Compression temps réel

6. Conclusion

Dans cet article, nous avons proposé une méthode pour la compression de données. Cette méthode est fondée sur la théorie des nombres premiers qui introduit une manière de compresser des données sans que ces données ne présentent des redondances. Ceci donne la possibilité d'améliorer le taux de compression en utilisant dans une première étape les différentes techniques de compression qui éliminent les redondances et dans une seconde étape l'algorithme décrit qui offre la faculté de compresser des flots de données ayant une entropie maximale.

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.
Hardy,Wright,(1980). *An introduction to the theory of numbers*. USA : Oxford University Press.
Lehning, Jakubowicz, (1982). *Mathématiques par l'informatique individuelle, tome 1* : le basic - arithmétique - cryptographie – équations. Paris: Edition Masson.
Arnaudiès, Fraysse, (1977). *Cours de mathématiques, tome 1* : algèbre. Paris: Edition Dunod.
D. Salomo, (1997). *Data Compression -The Complete Reference*. Berlin: Edition Springer.