18997 DÉCELLAS Victor

Algorithmes de compression de données sans perte

La compression de données est un domaine avec nombre d'applications pratiques mais aussi une grande richesse théorique. C'est également un thème à l'intersection entre les mathématiques, via la théorie de l'information, et l'informatique : deux sujets qui me passionnent.

La compression et la décompression de données sans perte consistent en une transformation de celles-ci pour les rendre moins lourdes. Il s'agit aussi de réduire la masse d'information tout en restant capable de restituer sans erreur le contenu initial.

Positionnement thématique (ÉTAPE 1) :

- *INFORMATIQUE (Informatique Théorique)*
- *INFORMATIQUE (Informatique pratique)*
- *MATHEMATIQUES (Mathématiques Appliquées)*

Mots-clés (ÉTAPE 1) :

Mots-clés (en français)	**Mots-clés (en anglais)**
compression	*compression*
théorie de l'information	*information theory*
entropie	*entropy*
codage arithmétique	*arithmetic coding*
dictionnaire	*dictionary*

Bibliographie commentée

La question de la transmission de nombreux fichiers à travers des canaux à débit limité et du stockage de l'information font qu'il est essentiel de pouvoir compresser des données le plus efficacement possible, que ce soit en mémoire gagnée ou en temps. De ce fait, nombre de technologies ont été mises en place pour répondre à ce besoin.

Ces algorithmes sont classés en deux grandes catégories : avec perte et sans perte. Ce TIPE se concentrera sur la deuxième classe, bien que les deux domaines connaissent une recherche florissante. Par exemple, des LLMs (Large Language Model) ont montré avoir des capacités de compression meilleures que les algorithmes "lossless" standards, tels que PNG pour les images

ou FLAC pour l'audio. [1] Notons que même si les algorithmes avec pertes aboutissent à des fichiers moins lourds, l'altération des données, même mineure, reste inacceptable dans certains cas, comme les fichiers de texte ou les exécutables.

Un simple argument de cardinalité prouve qu'il est impossible d'avoir un algorithme de compression "parfait" qui compresse strictement et sans perte toute chaîne de caractères. En fait, si une chaîne est strictement compressée par un algorithme sans perte, il en existe au moins une autre dont la version compressée est strictement plus longue que sa version initiale. [2] Un résultat plus fort implique que peu importe l'algorithme choisi, il est impossible de comprimer de manière optimale toutes les chaînes de caractères. C'est-à-dire qu'il existe toujours des chaînes comprimées plus efficacement par un autre algorithme que celui que l'on utilise. [4] [5] Mais en pratique, chaque technologie se spécialise sur un type de fichier précis, qui a un format connu et surtout beaucoup de redondances attendues.

Ces techniques de codage sont étudiées via la théorie de l'information, une théorie mathématique qui permet l'étude rigoureuse de la compression de données, mais dont les champs d'applications sont loin de se limiter à ce seul domaine. Le premier théorème de Shannon, notamment, établit une borne inférieure à la longueur moyenne d'un message comprimé : l'entropie. Ce même théorème donne une majoration de la plus petite longueur moyenne atteignable, également définie en fonction de l'entropie. [2]

Ce théorème du codage de source est à l'origine d'une classe d'algorithmes de compression dits "entropiques", qui visent à s'approcher de l'entropie de la source. Les deux principaux algorithmes de ce type sont le codage de Huffman ainsi que le codage arithmétique. On sait que l'algorithme de Huffman est effectivement optimal si l'on se restreint à un encodage symbole par symbole, mais aussi que sans cette contrainte, il existe de meilleurs algorithmes, dans le sens où ils parviennent à avoir des messages compressés dont la longueur moyenne est plus proche de l'entropie. On peut citer le codage arithmétique, qui non seulement atteint l'entropie dans les cas où Huffman y parvient, mais peut aussi être arbitrairement proche de celle-ci. En encodant par blocs de symboles, on peut être arbitrairement proche de cette valeur en utilisant le codage de Huffman, mais cette technique est particulièrement lente à cause du nombre considérable de possibilités à prendre en compte.[3] Cependant, l'exécution de tout algorithme entropique nécessite une estimation des fréquences des caractères de la chaîne, qui est fournie au préalable ou calculée à la volée.

Une autre approche fréquemment utilisée est le codage par dictionnaire, qui consiste à chercher dans un string des mots enregistrés dans une structure de données adaptée et à substituer chaque occurrence d'un mot donné par une référence vers cette dernière. L'algorithme de Huffman bloc par bloc est bien un codage par dictionnaire, car chaque mot est remplacé par son chemin correspondant dans un arbre. Cependant, le dictionnaire est la plupart du temps dynamique, comme dans LZ77 et LZ78, c'est-à-dire que l'ensemble des mots enregistrés évolue au fil de la compression. [3]

Problématique retenue

Comment compresser de manière à la fois rapide et efficace des fichiers de formats divers tout en étant capable de reconstituer sans altération le contenu d'origine ?

Objectifs du TIPE du candidat

Je me suis fixé comme objectif d'implémenter une bibliothèque de compression de données sans perte, à la fois polyvalente et efficace. En plus de l'implémentation, il s'agit aussi d'étudier à la fois dans la théorie et la pratique les résultats donnés par ces algorithmes, que ce soit individuellement ou lorsqu'on les combine.

Références bibliographiques (ÉTAPE 1)

[1] EDWARDS BENJ : AI language models can exceed PNG and FLAC in lossless compression, says study *: https://arstechnica.com/information-technology/2023/09/ai-language-models-can-exceed-png-and-flac-in-lossless-compression-says-study/*

[2] THOMAS M COVER & JOY A THOMAS : Elements of Information Theory *: chapitre 5 : "Data Compression", chapitre 6 : "Gambling and Data Compression"*

[3] GUY E. BLELLOCH : Introduction to Data Compression

[4] PETER BRO MILTERSEN : Course notes for Data Compression - 2 Kolmogorov complexity

[5] PIERRE CAGNE : Complexité de Kolmogorov