TP1 : Comparaison et caractérisation de méthodes de codage

Présenté à

Hughes Perreault

Par Élisa Correia-Martins - 1880313

Dans le cadre du cours

INF8770 : Technologies multimédias

Département de génie chimique

Polytechnique Montréal 28 septembre 2020

Question 1

Formulez une liste d'hypothèses à tester pour caractériser la performance des deux méthodes. Expliquez vos différentes hypothèses.

Hypothèse 1 : La méthode du codage par paires d'octets donnera un code plus compact que la méthode du codage arithmétique si le message contient un petit nombre de symboles différents puisqu'il y a plus de chances d'avoir des paires de symboles qui se répètent.

Hypothèse 2 : La méthode du codage arithmétique donnera un code plus compact que la méthode par paires d'octets si le message contient un grand nombre de symboles différents puisque la méthode par paires d'octets est efficace dans le cas où des symboles consécutifs se répètent et qu'avec beaucoup de symboles différents, il y a moins de chance d'obtenir plusieurs fois deux symboles consécutifs.

Hypothèse 3 : La méthode du codage arithmétique sera plus rapide que la méthode par paires d'octets pour un message aléatoire de grande taille puisque le codage par paires d'octets nécessite de calculer à chaque itération la fréquence d'apparition d'une paire. Un message aléatoire de grande taille nécessitera un grand nombre d'itérations.

Hypothèse 4 : La méthode du codage arithmétique donnera un code plus compact que la méthode par paires d'octets si le message a une entropie basse puisque le codage arithmétique est un encodage de type entropique qui code les symboles les plus fréquents avec moins de bits.

Question 2

Décrivez en détails les expériences que vous allez réaliser pour vérifier les hypothèses formulées (bases de données utilisées, contenu des messages, critère d'évaluation, code informatique utilisé, etc.).

Critères d'évaluations :

1. Le **taux de compression** : $1 - \frac{T_c}{T_o}$ où $T_c$représente la taille du message compressé et T_o, la taille du message original. Plus le taux de compression est élevé, plus l'encodage est efficace. Ce critère sera utilisé pour confirmer ou infirmer les hypothèses 1, 2 et 4.
2. Le **temps d'exécution** en secondes pour un même message. Plus le temps d'exécution est bas, plus l'algorithme est efficace. Ce critère sera utilisé pour confirmer ou infirmer l'hypothèse 3.

Expérience #1 : Afin de tester l'hypothèse 1, dix messages composés de deux symboles différents seront testés. Les messages auront chacun une longueur différente entre 5 et 1000 caractères. Le

taux de compression des messages sera ensuite calculé et comparé pour chaque méthode à l'aide d'un graphique.

Expérience #2 : Afin de tester l'hypothèse 2, dix messages composés de plusieurs symboles différents seront testés. Les messages auront chacun une longueur différente entre 5 et 1000 caractères. Les symboles utilisés proviennent de la liste de caractères ASCII. Le taux de compression des messages sera ensuite calculé et comparé pour chaque méthode à l'aide d'un graphique.

Pour les expériences 1 et 2, plusieurs messages de tailles variables sont analysés plutôt qu'un pour éviter que les résultats soient dus au nombre de symboles total, puisque nous voulons plutôt étudier les variances des résultats selon le nombre de symboles uniques.

Expérience #3 : Afin de tester l'hypothèse 3, les temps d'exécution de plusieurs messages aléatoire de taille variable seront enregistrés. Ces messages auront une longueur allant de 2 à 1000 caractères pour les deux méthodes. Les temps d'exécutions seront comparés à l'aide d'un graphique. Plusieurs messages de tailles variables sont analysés plutôt qu'un pour observer l'évolution du temps d'exécution selon la taille et la méthode.

Expérience #4 : Afin de tester l'hypothèse 4, trois messages d'entropie basse seront encodés à l'aide des deux méthodes. Ces messages sont issus de textes retrouvés sur internet de tailles différentes. Le taux de compression sera calculé, puis les résultats seront comparés à l'aide d'un tableau pour déterminer quelle méthode est la plus efficace pour chacun des textes.

Code informatique utilisé :
Afin d'évaluer la performance des deux méthodes de codage, le code écrit par Guillaume-Alexandre Bilodeau retrouvé sur le site Moodle du cours sera utilisé. (https://github.com/gabilodeau/INF8770URL) Plus précisément les fichiers « Codage arithmetique. ipynb » et « Codage par paire d'octets.ipynb ».

Pour générer des messages aléatoires, la fonction «get_random_string» du site PYnative sera adaptée et utilisée. (https://pynative.com/python-generate-random-string/)

Pour calculer le temps d'exécution dans l'expérience 3, la librairie « time » de Python sera utilisée.

Pour enregistrer les résultats dans des fichiers «.csv», la librairie « csv » de Python sera utilisée.

Question 3

Décrivez les codes informatiques utilisés pour réaliser les expériences. Décrivez, si applicable, comment vous avez adapté les codes informatiques pour réaliser les expériences décrites à la question 2.

Les algorithmes du codage par paires d'octets (paires.py) et du codage arithmétique (arithmetique.py) prennent en entrée un message, compressent le message et retournent un objet contenant la longueur du message original en bits, la longueur du message codé en bits et le temps d'exécution en secondes.

Le code possède aussi un fichier « main.py» qui réalise les quatre expériences et s'occupe de générer les messages et les résultats dans des fichiers «.csv».

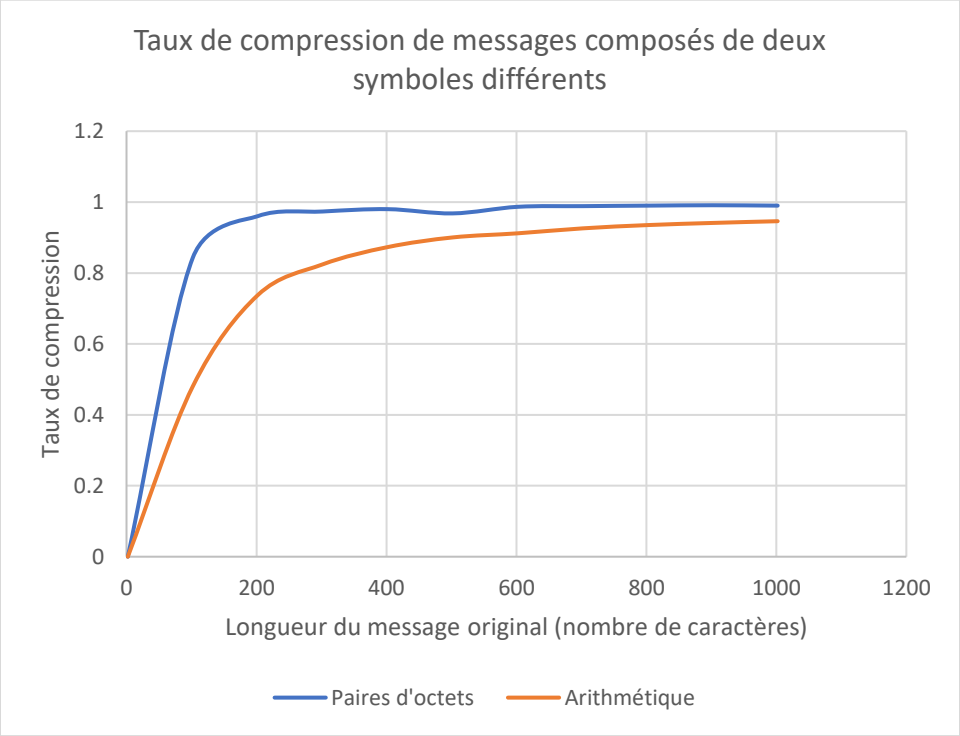
Afin d'adapter le code pour réaliser les expériences, il a fallu tout d'abord modifier le code pour enregistrer le temps d'exécution du début jusqu'à la fin de l'algorithme. Ensuite, il a fallu retourner un objet pour obtenir les résultats dans le format voulu.

Question 4

Analysez les résultats obtenus et mettez-les en relation avec les hypothèses de la question 1. Est-ce que les hypothèses sont supportées par les résultats ?

Expérience #1 :

Le graphique suivant montre les taux de compression de messages de différentes tailles composés des lettres « A » et « B ».

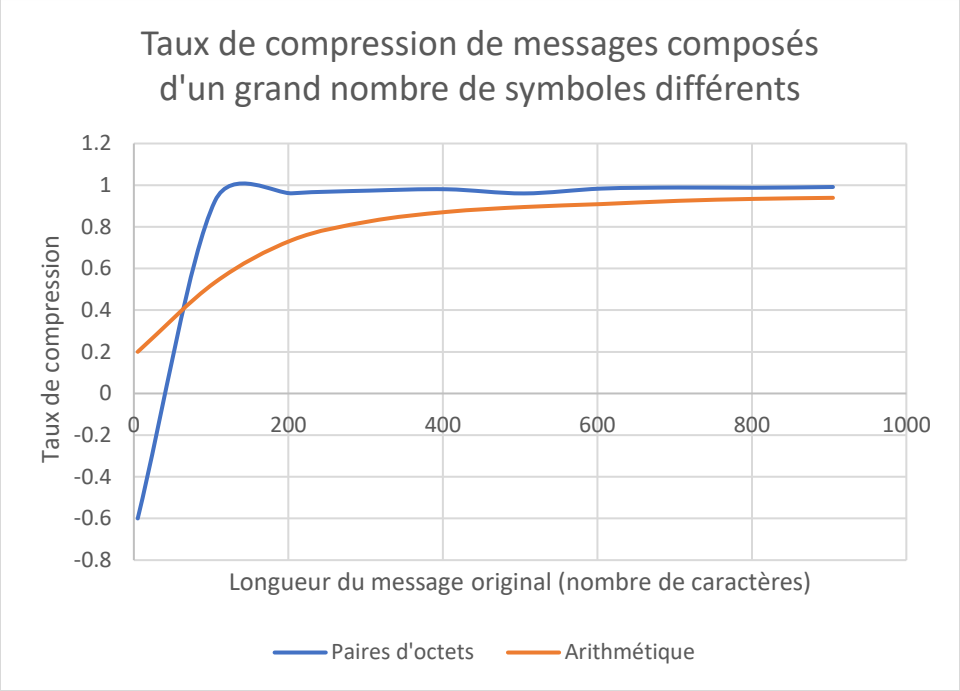


On peut observer que le codage par paires d'octets offre une meilleure compression de messages composés de peu de symboles différents que par la méthode du codage arithmétique. On peut

aussi observer une tendance, plus le message est long, plus l'écart entre les deux résultats est petit. L'hypothèse 1 est alors confirmée, le codage par paires d'octets offre une meilleure performance en termes de taux de compression pour des messages possédant peu de symboles uniques.

Expérience #2 :

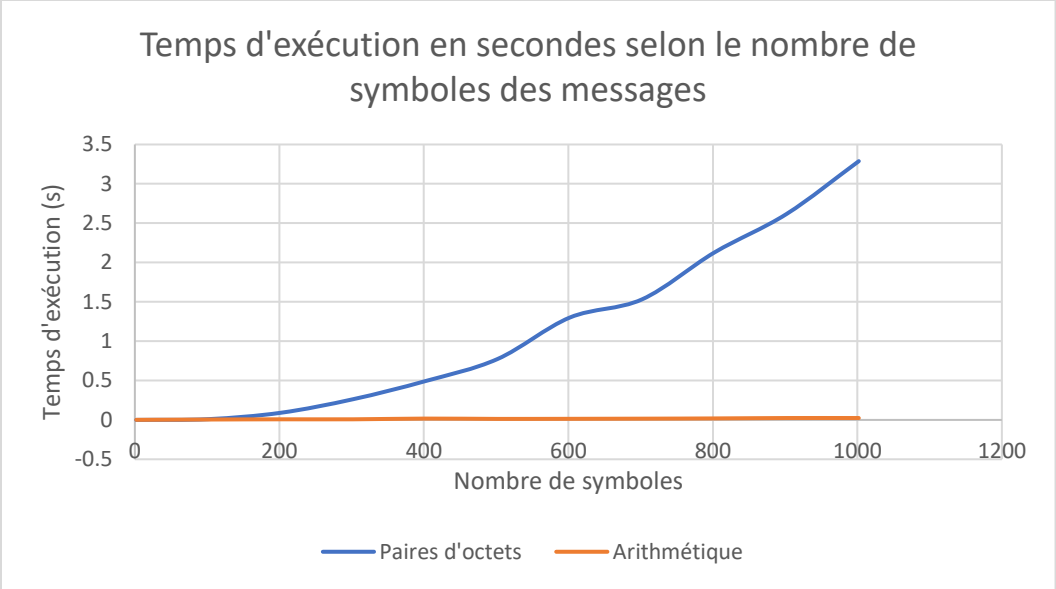
Le graphique suivant montre les résultats des taux de compression des messages de différentes tailles composés de plusieurs symboles différents.



On peut observer que le codage par paires d'octets offre généralement une meilleure compression de messages composés de beaucoup de symboles différents que la méthode du codage arithmétique sauf pour les très petits messages. On peut aussi observer la même tendance que dans la première expérience, plus le message est long, plus l'écart entre les deux résultats est petit à partir de 170 caractères environ. L'hypothèse 2 est alors infirmée. Cela peut être expliqué par l'entropie élevée des messages qui n'avantage pas le codage arithmétique puisqu'il s'agit de messages générés de manière complètement aléatoire.

Expérience #3 :

Le graphique suivant montre les temps d'exécution des deux méthodes de codage pour les mêmes messages allant de 2 à 1000 caractères aléatoire.



L'hypothèse 3 peut être confirmée. Le temps d'exécution de petits messages est plus ou moins pareil pour les deux méthodes. Cependant, plus le message est long, plus l'écart entre les deux méthodes est significatif. Le codage arithmétique est nettement plus rapide.

En revange, ce critère d'évaluation n'est peut-être pas le meilleur indicateur de performance, puisque divers facteurs peuvent expliquer pourquoi une telle méthode est plus rapide qu'une autre comme par exemple, une différence de qualité du code.

Expérience #4:

Titre du texte	Nombre de caractères	Taux de compression codage arithmétique	Taux de compression du codage par paires d'octets
In Every Girl There Is a Forest	464	**0.974138**	0.972845
The Disciple	907	**0.987688**	0.964719
The Mice	1182	**0.988663**	0.952623

L'hypothèse 4 peut être confirmée. Pour des messages de basse entropie, le codage arithmétique obtient des taux de compression légèrement plus élevé que pour le codage par paires d'octets. Le codage par paire d'octets est tout de même très efficace, puisque pour un message de basse entropie on peut tout de même espérer obtenir des paires d'octets qui se répètent. Par exemple, dans le texte « In Every Girl There Is a Forest », le mot « In» est répété 14 fois.