LOG8770 - Technologies multimédias

Remise initiale

Superviseur de projet:
Olivier Gendreau

Remis par :
Anthony Lachance - 1793057
Jonathan Moreau - 1798075

1 février 2019
École Polytechnique de Montréal

Question 1

Hypothèse 1: Le codage prédictif sera plus efficace pour un fichier où les données adjacentes évoluent de façon linéaire sur de grandes plages: L'erreur de la prédiction basée sur les voisins d'une donnée sera alors égale à la pente de cette évolution linéaire. La prédominance de la valeur de la pente dans les données sera alors comptabilisé dans le codage de Huffman qui suit le codage prédictif. Celui-ci sera alors codé avec un nombre minimal de bits, alors que le codage par paire d'octet va observer plusieurs paires d'octets différentes et aura un taux de compression moins élevé.

Hypothèse 2: Dans une image où les pixels sont identiques à plusieurs endroit différents dans la direction de détection des paires et sur plusieurs pixels, le codage par paires d'octet aura un taux de compression plus élevé que le codage prédictif puisque ces "lignes" de valeurs identiques seront compressés en un seul octet alors que le codage prédictif va réduire de nombre de bits pris par chaque pixels. Par contre, si les répétitions sont petites, le codage prédictif va l'emporter puisqu'il va pouvoir coder ces répétitions sur moins d'un octet (ce dont le codage par paire d'octet est incapable).

Hypothèse 3: Si un mauvais prédicteurs est choisis (c'est-à-dire un prédicteurs qui prédit des résultats rarement vraie et souvent éloignés du bon résultat), alors le codage prédictifs sera toujours moins bon que le codage par paires d'octets.

Question 2

Hypothèse 1:On va effectuer la compression sur des images présentant des gradients et comparer le taux de compression des deux méthodes.

Hypothèse 2:On va effectuer la compression sur des images présentants des pixels identiques, en ligne ou en carré à l'aide des 2 méthodes et comparer le taux de compression.

Hypothèse 3: On utilise les deux algorithmes de compression sur une image noir en prédisant du blanc. On vérifie ensuite en moyenne lequel des deux algorithmes produit les fichiers les moins lourds.

Question 3

Nous avons écrit deux fichiers en python, un pour chaque algorithme. Les deux commencent par importer une image png, la transformer en noir et blanc et convertir les valeurs en uint pour pouvoir travailler plus facilement avec. À la fin de l'exécution, les deux programmes affichent le taux de compression, c'est-à-dire le pourcentage de place qui a été gagné par rapport à l'original.

Le codage prédictif (suivie d'un huffman) est basé sur l'exemple de codage prédictif donné sur moodle. Il commence par calculer l'image prédite puis l'erreur. Le prédicteur de base prédit que le prochain pixel est le même que le précédent. Ensuite, un Huffman est appliqué pour réellement compressé les résultats. Ce dernier est aussi basé sur l'exemple de moodle.

Pour le codage par pairs d'octets, puisque l'exemple est pour un string, nous avons du le refaire pour l'adapter à la compression d'image. Il a principalement été reconstruit en utilisant son principe de fonctionnement vus en cours. Il a cependant été simplifié pour seulement donné la taille finale et ne donne pas un fichier décompressable à la fin. Pour ce faire il commence par calculer toutes les pairs possibles. Ensuite il compte les pairs qui se répètent et décide s'il peut encore gagner de la place en faisant des substitution. Ensuite, si il décide de continuer à substituer, il va chercher la pair la plus répétées et la substitue. Ce processus se répète jusqu'à ce qu'il n'y ait plus de pair répété plus de trois fois (si une pair se répète 3 fois, on économise pas de place à cause de la taille du dictionnaire).

Question 4

Hypothèse	Image	Taux de compression	
		Prédictif	Par pair
Hypothèse 1	gradient.png	87,5%	30,1%
	2gradient.png	87,4%	14,2%
Hypothèse 2	black_stripes.png	87,5%	99,9%
	stripes.png	87,5%	99,9%
	squares.png	87,4%	99,8%
Hypothèse 3	black.png	87,4%	99,9%

Sans surprise, le codage prédictif est globalement toujours assez bon. Ce n'est pas surprenant puisque que le codage prédictif avec le prédicteur que nous avons utilisé est un codage justement fait pour l'encodage d'image (contrairement au codage par pairs d'octet qui est plus adapté au texte).

Pour l'hypothèse 1, le codage par pair a d'ailleurs assez mal performé. En effet les gradients ne comptais pas tellement de pairs d'octets qui se répétaient puisque chaque ligne de l'image utilisait une couleur légèrement différente pour former le gradient. Au contraire, en utilisant notre prédicteur, l'erreur était globalement faible et a permis au Huffman de compresser beaucoup l'image. On remarquera qu'avec un tel gradient, si nous avions eu un prédicteur qui prévoyait l'évolution du gradient (ce qui est mathématiquement possible puisque qu'on peut représenter le gradient

comme une fonction), alors on aurait pu avoir un résultat encore meilleur avec une prédiction quasi parfaite.

Pour l'hypothèse 2, le codage prédictif est toujours efficace, en effet les pixels précédent étant souvent exactement de la même couleurs dans ces images, le codage avait encore une fois peu d'erreur. Cependant, le codage par pairs a tout de même dominé avec une compression énorme. En effet, avec autant de pairs qui se répètent, la taille finale est très proche de la taille du dictionnaire, qui reste finalement petite puisqu'il ne faut que 2 octets par couleurs présente dans l'image.

Pour l'hypothèse 3, sans surprise le codage par pair a encore superbement performé, pour les mêmes raisons que pour l'hypothèse 2, étant donné que l'image test ne comportait qu'une seule couleur, la taille finale n'était que de quelques octets. Cependant c'est le codage prédictif qui surprend en réussissant un taux de compression similaire aux expériences précédentes. Cela peut cependant s'expliquer. Même si le prédicteur se trompe tout le temps, tant que les mêmes valeurs d'erreurs ressortent régulièrement, Huffman sera capable d'en profiter pour compresser efficacement le fichier.