Devoir maison n°3 - À rendre le 25 Mars 2022
Compression de texte - partie 1

Notions abordées

- Compression de texte par ré-encodage des caractères
- Encodage sur l'alphabet $\{0,1\}$, à taille variable, non ambigu
- Représentation d'un codage par un arbre de décodage
- Codage et décodage d'un texte
- Score d'un encodage : taille moyenne du codage d'un caractère pour une distribution des caractères connue

Le code de ce projet peut être réalisé seul, en binôme ou en trinôme, mais les questions à rédiger sur papier (✍) doivent l'être individuellement. Les questions ✍ 🪜 peuvent être omises du rendu papier, qui se fera en classe. Le rendu de code se fera quant à lui sur cahier de prépa.

Toutes les fonctions doivent être commentées et **testées**. Elles doivent notamment être munies d'une description faisant suite à d'éventuelles hypothèses sur leurs arguments.
En OCaml, les commentaires s'écrivent comme suit `(*Commentaires*)`.

Rappel 1

*Un **alphabet** est un ensemble fini non vide. Si Σ est un alphabet, ses éléments sont appelés des **lettres**, et les séquence finies de lettres sont appelées des **mots** sur Σ.*

*La **longueur** d'un mot m est son nombre de lettres, on la notera $|m|$.*
*Le **mot vide**, noté ε, est le seul mot de longueur 0.*

Pour tout $n\in\mathbb{N}$, on note Σ^n l'ensemble des mots sur Σ de longueur n.
On note $\Sigma^* = \bigcup_{n\in\mathbb{N}} \Sigma^n$ et $\Sigma^+ = \bigcup_{n\in\mathbb{N}^*} \Sigma^n$

Rappel 2

Soit Σ est un alphabet. Soient $(u,v)\in\Sigma^*\times\Sigma^*$.
*La **concaténation** de u et v, notée $u.v$, est le mot constitué des lettres de u puis de celles de v.*
Ainsi, en notant $n=|u|, m=|v|$ et $w = u.v$, on a $\forall i\in[1\,..\,n+m]$, $w_i = \begin{cases} u_i & \text{si } i\leqslant n \\ v_{i-n} & \text{sinon} \end{cases}$

Rappel 3

Soit Σ est un alphabet. Soient $(u,v)\in\Sigma^*\times\Sigma^*$.
*On dit que u est **préfixe** de v, ce qu'on note $u\preccurlyeq v$, s'il existe $w\in\Sigma^*$, $v=u.w$.*
Ainsi, en notant $n=|u|$ et $m=|v|$, on a $u\preccurlyeq v$ ssi $m\geqslant n$ et $\forall i\in[1\,..\,n]$, $u_i = v_i$.

Encodage de caractères et de textes

On a vu que dans les langages C et OCaml, les caractères sont codés sur un octet, c'est-à-dire que chaque caractère est associé à l'une des 256 combinaisons de huit 0 et 1. Ainsi un texte de n caractères est encodé sur n octets, soit $8n$ bits. Cependant, si l'on ne considère qu'un nombre restreint de caractères, par exemple les 26 minuscules, les 26 majuscules, et une dizaine de signes de ponctuations, on pourrait se contenter d'un codage sur 6 bits qui offre 64 combinaisons de 0 et de 1 différentes, ainsi un texte de n caractères serait encodé sur $6n$ bits seulement. On peut même aller plus loin, en décidant de coder les caractères par des mots de 0 et de 1 de taille variable, en prenant soin de coder sur des mots courts les caractères les plus fréquents, afin de minimiser la taille du codage d'un texte.

Dans cette première partie, on s'intéresse d'abord à la construction d'un codage valide, c'est-à-dire qui permette d'encoder et décoder un texte sans erreur. On verra ensuite comment évaluer un codage valide, c'est-à-dire comment mesurer le taux de compression moyen qu'il offre pour des textes dont on connaît la distribution des caractères. En effet, la distribution des caractères dans une langue donnée est loin d'être uniforme, dans la langue français par exemple, on sait que le 'e' et le 'z' n'apparaissent pas du tout à la même fréquence. Enfin, nous nous intéresserons au problème d'optimisation qui consiste à trouver le meilleur codage valide, c'est-à-dire celui qui offre le meilleur taux de compression moyen.

Dans toute la suite, on note $\mathcal{C}$ l'ensemble fini des caractères que l'on cherche à encoder, autrement dit, les textes que l'on cherche à encoder sont des mots de $\mathcal{C}^*$. On note $\Sigma = \{0, 1\}$, ainsi le codage d'un caractère est un mot de Σ^*, et en tant que concaténation des codages de tous ses caractères, le codage d'un texte est lui aussi un mot de Σ^*.

Définition 4

Soit $\varphi \in \mathcal{F}(\mathcal{C}, \Sigma^+)$. On dit que φ est un ***codage*** *des caractères si $\bar{\varphi}$ est injective.*
De plus, si $im(\varphi) \subseteq \Sigma^n$ on dit que φ est un codage ***à taille fixe*** *sur n bits, sinon, on dit que c'est un codage* ***à taille variable*** *sur n bits pour $n = \min\{m \in \mathbb{N} \mid im(\varphi) \subseteq \bigcup_{k=1}^{m} \Sigma^k\}$.*

Pour toute fonction $\varphi \in \mathcal{F}(\mathcal{C}, \Sigma^+)$, on notera $\bar{\varphi}$ la fonction de $\mathcal{F}(\mathcal{C}^*, \Sigma^*)$ définie par $\bar{\varphi}(\varepsilon) = \varepsilon$ et $\forall c \in \mathcal{C}$, $\forall u \in \mathcal{C}^*$, $\bar{\varphi}(c.u) = \varphi(c).\bar{\varphi}(u)$. Autrement dit si φ est un codage des caractères, $\bar{\varphi}$ est le codage des textes associé, qui s'obtient par concaténation des codages des caractères du texte.

Exercice 1 Construire et représenter des codages non ambigus

Question 1 ✍ 🪜

Pour $\mathcal{C} = \{\texttt{a}, \texttt{b}, \texttt{c}\}$, la fonction φ^1 définie ci-contre est-elle un codage ? Est-ce un codage à taille fixe ou à taille variable, et sur combien de bits ? Si oui quel est alors le codage du texte `abba` ? Celui du texte `aacc` ?

$$\varphi^1 = \begin{pmatrix} \mathcal{C} & \to & \Sigma^+ \\ \texttt{a} & \mapsto & 00 \\ \texttt{b} & \mapsto & 01 \\ \texttt{c} & \mapsto & 10 \end{pmatrix}$$

Question 2 ✍ 🪜

Pour $\mathcal{C} = \{\texttt{a}, \texttt{b}, \texttt{c}\}$, la fonction φ^2 définie ci-contre est-elle un codage ? Est-ce un codage à taille fixe ou à taille variable, et sur combien de bits ? Si oui quel est alors le codage du texte `abba` ? Celui du texte `aacc` ?

$$\varphi^2 = \begin{pmatrix} \mathcal{C} & \to & \Sigma^+ \\ \texttt{a} & \mapsto & 0 \\ \texttt{b} & \mapsto & 01 \\ \texttt{c} & \mapsto & 10 \end{pmatrix}$$

Définition 5

Soit $\varphi \in \mathcal{F}(\mathcal{C}, \Sigma^+)$ un codage des caractères à taille variable sur n bits.
*On dit que φ est un **codage non ambigu** si $\bar{\varphi}$ est injective.*

Propriété 6

Soit $\varphi \in \mathcal{F}(\mathcal{C}, \Sigma^+)$ un codage des caractères de $\mathcal{C}$.
Si aucun mot de $im(\varphi)$ n'est préfixe d'un autre mot de $im(\varphi)$, alors φ n'est pas ambigu. [1]

Question 3 ✍

Démontrer la propriété 6. *On peut montrer la contraposée, c'est-à-dire montrer que l'ambiguïté de φ impose qu'il existe dans $im(\varphi)$ deux mots différents dont l'un est préfixe de l'autre.*

On a défini les nœuds d'un arbre binaire, comme les chemins admissibles pour cet arbre, et ses feuilles comme étant les nœuds dont le chemin ne pouvait être prolongé en un autre chemin admissible. En utilisant le vocabulaire des mots, cela signifie que l'ensemble des feuilles est un ensemble de mots sur Σ dont aucun n'est préfixe d'un autre nœud de l'arbre, et en particulier d'aucune feuille. De ce fait, en choisissant comme ensemble d'arrivée pour un codage φ les feuilles d'un arbre binaire non réduit à une feuille, on obtient nécessairement un codage non ambigu. On évite l'arbre réduit à une feuille, car dans cet arbre la seule feuille est la racine, et son chemin est donc le mot vide qu'on a exclu comme codage possible pour un caractère. Un tel codage φ se représente très facilement à condition que $im(\varphi)$ couvre toutes les feuilles : il suffit d'étiqueter chaque feuille par le caractère qu'elle encode.

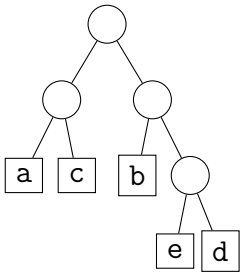
Question 4 ✍ 🪜

Dessiner l'arbre représentant le codage φ^3 défini ci-contre pour l'ensemble de caractères $\mathcal{C} = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}, \texttt{e}\}$.

$$\varphi^3 = \begin{pmatrix} \mathcal{C} & \to & \Sigma^+ \\ \texttt{a} & \mapsto & 000 \\ \texttt{b} & \mapsto & 001 \\ \texttt{c} & \mapsto & 11 \\ \texttt{d} & \mapsto & 01 \\ \texttt{e} & \mapsto & 10 \end{pmatrix}$$

Question 5 ✍ 🪜

Donner le codage φ^4 correspondant à l'arbre ci-contre.



Question 6 ✍

On définit un **arbre de décodage** comme étant une feuille étiquetée par un caractère ou un arbre constitué d'une racine sans étiquette et de deux arbres de décodage. Donner les règles de construction permettant de construire par induction l'ensemble des arbres de décodage, que l'on notera $\mathcal{A}_D$.

Question 7 💻

Définir en OCaml un type `arbre_d` pour les arbres de décodage.
Définir `ad3` et `ad4` les objets de type `arbre_d` représentant les arbres associés à φ^3 et φ^4 respectivement.

Question 8 ✍

Définir par induction la fonction ch_f qui associe à un arbre de décodage l'ensemble des chemins de ses feuilles. *Vous pouvez vous inspirer de ce qui a été fait en cours pour les arbres binaires.*

Question 9 💻

On choisit de représenter les mots de Σ^* par des listes de booléens, `false` codant pour `0` et `true` codant pour `1`. Ainsi le mot vide est encodé par la liste vide, et 001 par `[false; false; true]` par exemple. Définir en OCaml un type `codage` pour les mots sur Σ^*.

[1] On appelle parfois un tel codage "code préfixe", ou *"prefix-free code"* en anglais.

Question 10 💻
Donner une fonction de signature `char_from_cod (a:arbre_d)(co:codage) : char` qui calcule le caractère en étiquette de la feuille de chemin `co` dans `a` si `co` est bien le chemin d'une feuille dans `a`, et qui lève une exception `CodeNonValide` sinon. *On pourra tester les cas sans exception de cette fonction dans un jeu de test utilisant les arbres de décodage* `ad3` *et* `ad4`.

Question 11 💻
Donner une fonction de signature `cod_from_char_naif (a:arbre_d)(c:char) : codage` qui calcule le codage indiqué par `a` pour le caractère `c`, c'est-à-dire le chemin dans `a` de la feuille qui encapsule le caractère `c`, en supposant qu'il existe bien une telle feuille. Dans le cas contraire, cette fonction lèvera une exception `CaractereNonPresent`.

Question 12 ✍
La **taille** d'un arbre de décodage est son nombre de nœuds (internes) et de feuilles. La **hauteur** d'un arbre de décodage est 0 s'il s'agit d'une feuille, et $1+\max(h(a),h(b))$ s'il s'écrit $(N,\_,a,b)$. Par exemple l'arbre de la question 5 est de hauteur 3, et de taille 9 car il a 5 feuilles de 4 nœuds internes. Quelle est la hauteur maximum (resp. minimum) d'un arbre de décodage à n feuilles ? Quelle est la taille d'un arbre de décodage à n feuilles ?

Question 13 ✍
En déduire de quel ordre serait la complexité pire cas de la fonction `char_from_cod` en fonction de la taille de l'arbre de décodage qu'elle prend en paramètre ?

L'opération de recherche d'un caractère est celle qui permet d'obtenir le codage de ce caractère. C'est donc une opération essentielle lors du codage d'un texte. La fin de cet exercice, constituée de questions bonus, est dédiée à l'amélioration de la complexité de cette opération.

Pour cela, on va créer une structure de données adaptée à la recherche de caractères, en faisant l'hypothèse que l'ensemble $\mathcal{C}$ des caractères encodés est muni d'une relation d'ordre totale. Plus précisément, cette structure permettra de représenter un sous-ensemble A de $\mathcal{C}\times\Sigma^*$ dont la première composante constitue une clé, (*i.e.* $\forall(c,u)\in\mathcal{C}\times\Sigma^*, \forall(c',u')\in\mathcal{C}\times\Sigma^* \, c=c' \Rightarrow u=u'$) de manière à ce que la recherche par clé soit efficace, et qu'on puisse donc atteindre efficacement le couple qui associe à un caractère son codage.

Question 14 ✍ ∗
Quelle structure de données vue en cours serait adaptée ?

Question 15 💻 ∗
Définir un type `strucure_e` pour la structure de données proposée à la question précédente, ainsi que les éventuelles fonctions nécessaires à sa construction et à sa à sa manipulation.

Question 16 💻 ∗
Définir une fonction qui construit un objet de type `structure_e`, adapté à l'encodage, à partir d'un arbre de décodage `a`. Plus précisément, l'objet construit doit permettre de calculer rapidement, pour chaque caractère en feuille dans `a`, le chemin de cette feuille qui est l'encodage de ce caractère.

Question 17 💻 ∗
Donner une fonction de signature `cod_from_char (s:structure_e)(c:char): codage` qui calcule l'encodage du caractère `c` selon `s`.

Exercice 2 Codage et décodage

Question 1 ✍ 🪜
Procéder manuellement au codage de la chaîne de caractères `"abcade"` pour les codages φ^3 et φ^4, autrement dit calculer $\bar{\varphi}^3$(`"abcade"`) et $\bar{\varphi}^4$(`"abcade"`).

Question 2 🖳
Donner une fonction de signature `code (a:arbre_e)(s:string) : codage` qui calcule le codage de la chaîne de caractères `s` selon le codage représenté par `a`. *Vous prendrez garde au sens de lecture et au sens d'écriture.*

Question 3 ✍ 🪜
Procéder manuellement au décodage de `"000110110"` et `"00100000100011"` pour le codage φ^3, en utilisant comme support l'arbre de décodage associé proposé à la question 4 de l'exercice 1. Que dire du décodage de `"0001"` ?

Question 4 ✍ puis 🖳
On rappelle que le codage d'une chaîne de caractères est la concaténation des codages de ses caractères. Pour un codage à taille fixe, on sait a priori séparer un codage de chaîne en codages des différents caractères, mais pour un codage à taille variable ce n'est pas le cas. Néanmoins, comme l'a montré la question précédente, on est capable de décoder le codage d'une chaîne.
En effet, on constate la délimitation entre le codage du premier caractère et celui du deuxième lorsque, en se déplaçant dans l'arbre de décodage suivant le début du codage, on atteint une feuille. Dans ce cas on atteint une feuille avant la fin du codage, mais ce n'est pas une erreur.

Dans la fonction `char_from_cod_1` proposée précédemment, repérer quel déclenchement d'exception `CodeNonValide` correspond à ce cas.

Définir une fonction de signature `char_from_cod_2 (a:arbre_d) (co:codage) : char*codage` qui lit `co` jusqu'à trouver une feuille dans `a`, et qui retourne alors cette feuille et le reste du codage dans un couple. Dans le cas où le codage ne mène pas à une feuille, la fonction déclenchera l'exception `CodeNonValide`.

Question 5 🖳
Donner une fonction `decode` de signature `decode (a:arbre_d)(co:codage) : string` qui calcule la chaîne de caractères encodée par un codage `co` selon le codage représenté par `a` en utilisant la fonction `char_from_cod_2`.

Exercice 3 Score d'un codage

Dans cet exercice, on souhaite évaluer les codages à l'aune de la compression, on s'intéresse donc à la taille du codage d'un texte, relativement à la taille de ce texte. En effet, la taille du codage d'un caractère étant variable, la taille du codage d'un texte ne dépend pas seulement de son nombre de caractères, mais aussi de la distribution de ces caractères, c'est-à-dire du nombre d'occurrences de chaque caractère [2]. Pour un texte donné, ou du moins pour une distribution donnée, on préfère donc un encodage dans lequel les caractères les plus fréquents ont le codage le plus petit, quitte à ce que les caractères les moins fréquents aient un codage plus long.
On s'intéresse donc au calcul d'un score pour évaluer un codage pour une distribution donnée. Dans la mesure où il s'agit seulement d'une procédure d'évaluation d'un encodage, et pas d'une procédure effective lors d'un encodage, on ne s'attachera pas à faire ce calcul de manière efficace.

Définition 7

Soit $f \in \mathcal{F}(\mathcal{C}, \mathbb{R}^+)$.
*On dit que f définit **une distribution** des caractères de $\mathcal{C}$ si $\sum_{c\in\mathcal{C}} f(c) = 1$.*
*Dans ce cas, la **fréquence** d'un caractère $c \in \mathcal{C}$ est $f(c)$.*

Définition 8

Soit $f \in \mathcal{F}(\mathcal{C}, \mathbb{R}^+)$ une distribution de caractères. Soit $\varphi \in \mathcal{F}(\mathcal{C}, \Sigma^*)$ un codage.
*On définit le **score** de φ (pour la distribution f) comme étant la taille moyenne du codage d'un caractère, c'est-à-dire $\sum_{c\in\mathcal{C}} f(c)\,|\varphi(c)|$.*

Question 1 ✍
Calculer le score de φ^3 et φ^4 pour la distribution uniforme u sur $\mathcal{C} = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}, \texttt{e}\}$, *i.e.* $\forall c \in \mathcal{C}, u(c) = \frac{1}{5}$.
Calculer le score de φ^3 et φ^4 pour la distribution f donnée par $f(\texttt{a}) = 0.2$, $f(\texttt{b}) = f(\texttt{c}) = f(\texttt{d}) = 0.1$, et $f(\texttt{e}) = 0.5$. *Le résultat peut-être laissé sous forme de fraction.*

Question 2 💻
On représente une distribution par une liste de couples associant un caractère à sa fréquence.
Définir le type `distribution` en OCaml.

Question 3 💻 🪜
Donner une fonction `taille_codage` qui prend en argument un arbre de décodage `a` et un caractère `c` présent dans `a`, et qui calcule la longueur du codage de `c` selon `a`. Bien que la taille d'un codage soit toujours un entier, cette fonction rendra la longueur sous la forme d'un flottant en vue de simplifier les calculs ultérieurs.

Question 4 💻
Donner une fonction `score_codage` qui prend en argument un arbre de décodage `a` et une distribution `d` dont les caractères sont présents dans `a`, et qui calcule la longueur moyenne, au sens des fréquences données par `d`, de la taille du codage d'un caractère. *Attention aux tests d'égalité sur les `float` dans le jeu de tests.*

Question 5 ✍
Expliquer comment, étant donné un texte et un codage, on peut calculer le taux de compression, c'est-à-dire le ratio entre la taille du codage de ce texte, et la taille du codage si chaque caractère est encodée sur un octet, *i.e.* par un mot de Σ^8.

[2] En réalité, sans connaître exactement le nombre d'occurrences de chaque caractère, il suffit de connaître le nombre de caractères pour chaque taille de codage.

Question 6 ✍ 🪜

Sans connaître a priori le texte à compresser, on peut se baser sur la fréquence des lettres dans une langue. Dans la langue française par exemple, la lettre `e` est de loin la plus présente (12%), presque deux fois plus que le `a`... Dans l'arbre de décodage associé à un bon codage pour la langue française (*i.e.* de faible score), quelle est selon vous la place de la feuille contenant `e` ?
Si cet arbre de décodage est construit de bas en haut, c'est-à-dire par rassemblements successifs d'arbres sous une même racine, les arbres initiaux étant réduits à des feuille, à quel moment la feuille contenant `e` devrait-elle être ajoutée ? *On demande une réponse intuitive, des explications, pas de preuve.*

Question 7 ✍ 🪜

Grâce à la remarque faite à la question précédente, proposer un meilleur codage que φ^3 et φ^4 pour la distribution f sur $\mathcal{C} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}\}$. *Il peut être intéressant pour la suite d'essayer de construire l'arbre de décodage associé en commençant par les feuilles.*

On étend la définition de score à un codage partiel. Si $f \in \mathcal{F}(\mathcal{C}, \mathbb{R}^+)$ est une distribution sur $\mathcal{C}$, et $\varphi' \in \mathcal{F}(\mathcal{C}', \Sigma^*)$ un codage sur $\mathcal{C}' \subseteq \mathcal{C}$, on définit le score de φ' comme étant $\sum_{c \in \mathcal{C}'} f(c)\,|\varphi'(c)|$.
De plus on parlera du score d'un arbre de décodage pour désigner le score du codage associé.

Question 8 ✍

On considère $\mathcal{C}$ un ensemble de caractères muni d'une distribution f. Si a est un arbre de décodage sur $\mathcal{C}' \subseteq \mathcal{C}$ non réduit à une feuille, comment exprimer son score à partir du score de ces sous-arbres gauche et droit ?

Question 9 ✍ ∗

En utilisant les remarques et observations des dernières questions, proposer un algorithme pour construire par les feuilles un arbre de décodage de score minimal, la distribution des caractères étant connue. *Il n'est pas demandé d'implémenter un tel algorithme, on attend seulement une description en français de la procédure, ou du pseudo-code. Il peut être utile de tester l'algorithme proposé sur les exemples ci-dessus, ou sur la distribution uniforme sur un ensemble de caractères plus grand.*