TD 6 : Codage

1 Codes uniquement déchiffrables

Cette partie propose une démonstration de la correction l'algorithme de Sardinas & Patterson qui décide en temps polynomial si un code est uniquement déchiffrable.

Si $u, v \in \Sigma^*$, on définit $u - v = \{u' \in \Sigma^* \mid u = v \cdot u'\}$. Ainsi, $u - v$ est réduit à $\emptyset$ ou un élément. (Par exemple, $abc - a = \{bc\}$, $ab - ba = \emptyset$.)
Étant donné $S \subseteq \Sigma^*$, on définit

$$T_0 = \bigcup_{\substack{u,v \in S \\ u \neq v}} u - v$$

et on définit T comme étant le plus petit ensemble qui contient T_0 et qui satisfait l'inégalité

$$\bigcup_{s \in S, v \in T} ((s - v) \cup (v - s)) \subseteq T$$

Question 1 : L'objectif de cette question est de montrer que le code défini par S, $C : \mathcal{X} \to S$, est uniquement déchiffrable ssi T ne contient pas le mot vide. Pour tout $i \geq 1$, on pose :

$$T_i = T_{i-1} \cup \bigcup_{u \in S, v \in T_{i-1}} ((u - v) \cup (v - u))$$

Il est facile de voir que $T = \bigcup_i T_i$ et que les T_i ne contiennent que des suffixes de mots de S. En outre, puisque $T_i \subset T_{i+1}$ pour $i \geq 0$, il existe $n \geq 0$ tel que $T = T_n = T_{n+1}$.

Démontrer les deux lemmes suivants :

(i) S'il existe $u \in T$ et $u_1, \ldots, u_k, v_1, \ldots, v_\ell \in S$ tels que $u_1 u_2 \cdots u_k = u v_1 v_2 \cdots v_\ell$, alors $\varepsilon \in T$.

(ii) Pour tout i, s'il existe $u \in T_i$ et $u_1, \ldots, u_k, v_1, \ldots, v_\ell \in S$ tels que $u u_1 \ldots u_k = v_1 \ldots v_\ell$, alors il existe $v \in T_0$ et $u'_1, \ldots, u'_n, v'_1, \ldots, v'_m \in S$ tels que $v u'_1 \ldots u'_n = v'_1 \ldots v'_m$.

Conclure la démonstration.

Question 2 : Parmi les ensembles S suivants, lesquels définissent des codes uniquement déchiffrables ?

$S_0 = \{0, 10, 11\}$ $\quad$ $S_1 = \{0, 01, 11\}$
$S_2 = \{0, 01, 10\}$ $\quad$ $S_3 = \{0, 01\}$
$S_4 = \{00, 01, 10, 11\}$ $\quad$ $S_5 = \{110, 11, 10\}$
$S_6 = \{110, 11, 100, 00, 10\}$

Question 3 : Donner un algorithme polynomial pour décider si un ensemble $S \subseteq \Sigma^*$ définit un code uniquement déchiffrable. Estimer la complexité de l'algorithme.

2 Codage de flux de données

2.1 Découpage d'un flux infini

On veut coder un mot infini $w \in \Sigma^\omega$ par une suite de mots dans $S \subset \Sigma^+$. Afin de coder tous les mots possibles, sans ambiguïté, on exige de S les deux propriétés suivantes :

(A) Complétude : tout mot infini $w \in \Sigma^\omega$ admet un préfixe $v \in S$

(B) S est "ω-uniquement déchiffrable" : Si $(u_i)_i$ et $(v_i)_i$ sont deux suites de mots de S telles que $u_1 u_2 \ldots = v_1 v_2 \ldots$, alors $\forall i\ u_i = v_i$.

Question 4 : Montrer que S définit un code préfixe, c'est-à-dire que pour tout u et v dans S, si u est un préfixe de v alors $u = v$.

Dans la suite, on va s'intéresse au codage d'un flux de données décrit par une séquence $(U_i)_{i>0}$ de lettres aléatoires ($U_i \in \Sigma$). La séquence est supposée infinie, et nous souhaitons pouvoir coder, puis décoder, le message au fur et à mesure.

2.2 Code de Elias

Question 5 : On note $B_0(n) \in \{0,1\}^+$ l'écriture en base 2 de l'entier $n \geq 1$ en mettant le bit de poids faible à droite, le bit le plus à gauche étant 1. Exprimer $|B_0(n)|$ et donner un équivalent (quand n tend vers $+\infty$.) Est-ce que B_0 est un code préfixe ?

Question 6 : Mêmes questions pour $B_1(n) = 0^{|B_0(n)|-1} \cdot B_0(n)$ (i.e., le mot formé de $k = |B_0(n)| - 1$ chiffres 0 suivis de $B_0(n)$.)

Question 7 : (**DM**-1) Mêmes questions pour $B_2(n) = B_1(|B_0(n)|) \cdot B_0(n)[2, -]$.

2.3 Codage par rang

On suppose dans cette sous-partie ne pas disposer des fréquences d'apparition des lettres données en entrée. Nous allons concevoir un algorithme de compression en ligne, dont le codage change au cours du temps, afin de s'adapter aux fréquences de lettres effectivement constatées.

Pour cela, on conserve à chaque instant, la liste des lettres ordonnée par date de dernière apparition. On note $W_k = x_1 \ldots x_{|\Sigma|} \cdot U_1 \ldots U_k$ le mot lu en entrée à l'instant k, préfixé par toutes les lettres de l'alphabet d'entrée. Ainsi, toute lettre apparaît au moins une fois dans W_k.

Question 8 : Soit $x \in \Sigma$ une lettre quelconque, et un indice $k > 0$. On note

$$N_k[x] = 1 + \min\{|P| \mid P \subseteq \Sigma \quad \wedge \quad W_k \in \Sigma^* \cdot x \cdot P^*\}$$

Expliquer ce que représente le tableau N_k, et donner un algorithme calculant N_{k+1} à partir de N_k et de U_{k+1} en temps linéaire ($O(|\Sigma|)$ opérations).

Question 9 : On se donne une fonction $C : [1, |\Sigma|] \to \{0,1\}^+$ injective. On code la k-ième lettre lue U_k grâce au mot $V_k = C(N_{k-1}[U_k])$. Donner un algorithme *en ligne* lisant en entrée les mots $V_1, V_2 \ldots$ et écrivant en sortie les lettres $U_1, U_2 \ldots$.

Question 10 : On suppose désormais que l'algorithme lit le mot infini "$V_1 V_2 \ldots$" lettre par lettre. Quelle hypothèse supplémentaire doit-on faire sur la fonction C ?

Question 11 : Soit $\Delta_k = \min\{i \geq 1 \mid W_k[|W_k| - i] = U_k\}$. Que représente Δ_k ? Montrer que $N_{k-1}[U_k] \leq \Delta_k$.

Question 12 : On suppose les U_i indépendants et identiquement distribués. On fixe une lettre $u \in \Sigma$. Calculer la limite de $\mathsf{E}(\Delta_k \mid U_k = u)$.

Question 13 : On suppose $C = B_1$. Montrer que la longueur d'une lettre codée vérifie :

$$\limsup_{k\to\infty} \mathsf{E}(|V_k|) \leq 2\mathsf{H}(U) + 1$$

Question 14 : (**DM**-2) On suppose cette fois $C = B_2$. Montrer que :

$$\limsup_{k\to\infty} \mathsf{E}(|V_k|) \leq \mathsf{H}(U) + 2\log(\mathsf{H}(U) + 1) + 1$$

Question 15 : Comment améliorer ce taux de compression ? À quel coût ?