Les plus longs cycles et chemins

Un **graphe non orienté** G est un couple (V, E) où V est un ensemble de sommets et E un ensemble d'arêtes. Notons $n = |V|$, $m = |E|$. Les sommets u et v sont dits **voisins** s'il y a une arête notée $\{u, v\}$ entre u et v. Nous considérons les notations suivantes :

1. Le **degré** de v dans G, que l'on note $d_G(v)$, est le nombre de voisins de v.
2. Un **chemin (simple)** C allant de s à t est une suite $u_0, u_1, \ldots, u_\ell$ de sommets telle que $u_0 = s$, $u_\ell = t$, $(u_{i-1}, u_i) \in E$ pour tout $1 \leq i \leq \ell$ et telle que les sommets apparaissent une seule fois dans le chemin (sauf éventuellement ses extrémités). Un **cycle** est un chemin dont ses extrémités sont identiques ($u_0 = u_\ell$).
3. Un chemin (resp. cycle) est **hamiltonien** s'il traverse tous les sommets du graphe exactement une fois.

Question 1. Décrire un algorithme qui, étant donné un graphe G et une suite de sommets C, retourne un booléen qui indique si C est un cyle hamiltonien. Évaluer sa complexité.

Question 2. Décrire un algorithme qui, étant donné un graphe G, retourne un booléen qui indique si le graphe G respecte la propriété suivante : la somme des degrés de toute paire de sommets u et v (avec $u \neq v$) non voisins vaut au moins n, c'est-à-dire $d_G(v)+d_G(u) > n$. Évaluer sa complexité.

Dans la suite, nous considérons un graphe non orienté de $n \geq 3$ sommets tel que la somme des degrés de toute paire de sommets non voisins est strictement supérieure à n.

Question 3. Soit v un sommet du graphe. Montrer que $n - 1 \geq d_G(v) \geq 2$.

Question 4. Montrer que G est connexe.

Question 5. Soit $U = u_0, u_1, \ldots, u_{n-1}, u_0$ une suite de $n + 1$ sommets contenant tous les sommets de G. Supposons que U n'est pas un cycle dans G.

Notons $a(U) = |\{i : \{u_i, u_{i+1}\} \in E \text{ et } 0 \leq i < n\}|$.

Question 5.1 Supposons que u_ℓ et $u_{\ell+1}$ ne sont pas voisins dans G. Montrer qu'il existe une suite $U' = u'_0, u'_1, \ldots, u'_{n-1}, u'_0$ de $n+1$ sommets contenant tous les sommets de G telle que $a(U') > a(U)$. *Indication : considérer l'ensemble des sommets*

$$X = \{u_{i+1} : i \neq \ell - 1 \wedge \{u_\ell, u_i\} \in E\}$$

Question 5.2 Décrire un algorithme qui, étant donnée une suite $U = u_0, u_1, \ldots u_{n-1}, u_0$ contenant tous les sommets de G telle que $\{u_0, u_1\} \notin E$, retourne une suite $U' = u'_0, u'_1, \ldots, u'_{n-1}, u'_0$ contenant tous les sommets de G telle que $a(U') > a(U)$. Évaluer sa complexité.

Question 6. Montrer que le graphe G admet un cycle hamiltonien.

Question 7. Décrire un algorithme qui, étant donné un graphe G, retourne un cycle hamiltonien. Évaluer sa complexité.

Un **graphe orienté** G est un couple (V, A) où V est un ensemble de sommets et $A \subseteq V \times V$ est un ensemble d'arcs. Rappelons les notations dans le contexte des graphes orientés :

1. L'arc de u vers v que l'on notera par $(u \to v)$ est un arc **entrant** dans v.
2. Un chemin C allant de s à t est une suite $u_0, u_1, \ldots, u_\ell$ de sommets telle que $u_0 = s$, $u_\ell = t$, $(u_{i-1} \to u_i) \in A$ pour tout $1 \leq i \leq \ell$. Un **cycle** est un chemin dont ses extrémités sont identiques ($u_0 = u_\ell$).

Nous considérons un graphe orienté sans cycle sans contrainte sur les degrés. Soit G un graphe orienté sans cycle.

Question 8. Montrer que G possède au moins un sommet u_0 sans arc entrant.

Question 9. Décrire un algorithme qui, étant donné un graphe orienté sans cycle G, retourne un chemin hamiltonien (s'il existe). Évaluer sa complexité.

Un chemin C de s à t est dit **un plus long chemin** s'il a le plus grand nombre d'arcs parmi tous les chemins allant de s à t.

Question 10. Soit $C = u_0, u_1, \ldots, u_\ell$ un plus long chemin allant de u_0 à u_ℓ dans un graphe orienté sans cycle. Montrer que les sous-chemins de C allant de u_0 à u_i (avec $0 \leq i \leq \ell$) et allant de u_i à u_ℓ sont aussi des plus longs chemins. Est-ce le cas dans un graphe orienté avec cycle ?

Question 11. Donner la formule de récurrence qui permet de calculer la longueur d'un chemin le plus long finissant par u.

Question 12. Donner un algorithme qui retourne la longueur d'un chemin le plus long finissant par u, pour tout sommet u du graphe. Évaluer la complexité.

Question 13. Adapter l'algorithme précédent pour qu'il retourne le plus long chemin dans le graphe. Évaluer la complexité.

Préserver la régularité coûte que coûte

Soit Σ un alphabet fini et non vide. La longueur d'un mot $w \in \Sigma^*$ est notée $|w|$. Un **automate fini** $\mathcal{A}$ sur l'alphabet Σ est la donnée d'un tuple $\mathcal{A} = \langle Q, \delta, I, T \rangle$ où Q est un ensemble fini d'états, $\delta \subseteq Q \times \Sigma \times Q$ est l'ensemble des transitions, $I \subseteq Q$ est l'ensemble des états initiaux et $T \subseteq Q$ est l'ensemble des états terminaux. La transition $(p, \sigma, q) \in \delta$ est notée par $p \xrightarrow{\sigma} q$. Un **calcul** c de $\mathcal{A}$ est une séquence finie de transitions qui forment un chemin dans le graphe de $\mathcal{A}$, $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$: on note un tel calcul $c = q_0 \xrightarrow{\sigma_1\sigma_2\cdots\sigma_n} q_n$. L'**étiquette** de c est le mot $\sigma_1\sigma_2\cdots\sigma_n$ de Σ^*. Le calcul est **acceptant** si $q_0 \in I$ et $q_n \in T$. Le **langage** de $\mathcal{A}$ est le sous-ensemble $\mathcal{L}(\mathcal{A})$ de Σ^* qui contient l'ensemble des étiquettes des calculs acceptants de $\mathcal{A}$. Un tel langage est dit **régulier**. L'automate $\mathcal{A}$ est dit **déterministe** si I possède un unique état et pour tout $p \in Q$ et $\sigma \in \Sigma$, il existe au plus un état $q \in Q$ tel que $(p, \sigma, q) \in \delta$.

Ce sujet propose de caractériser les applications $f\colon \mathbf{N} \to \mathbf{N}$ telle que pour tout langage régulier, le langage $L_f = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } uv \in L\}$ est aussi régulier. On dit qu'une telle fonction **préserve la régularité**.

1. Montrer que la fonction identité préserve la régularité, c'est-à-dire que si L est un langage régulier, alors $L' = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |u| = |v| \text{ et } uv \in L\}$ est régulier.

2. Montrer que toute fonction affine, de la forme $f\colon n \mapsto an + b$ avec $a, b \in \mathbf{N}$, préserve la régularité. *On pourra commencer par considérer les cas particuliers où $a = 0$, puis $b = 0$.*

Pour généraliser davantage l'étude de la préservation de la régularité, introduisons la notion de **matrice de transitions** d'un automate $\mathcal{A} = \langle Q, \delta, I, T \rangle$: il s'agit de la matrice carrée booléenne $\Delta = (\Delta_{p,q})_{p,q \in Q} \in \mathbf{B}^{Q^2}$ indexée par les états de Q, avec $\mathbf{B} = \{\top, \bot\}$, tel que $\Delta_{p,q} = \top$ (vrai) s'il existe un symbole $\sigma \in \Sigma$ avec une transition $p \xrightarrow{\sigma} q$ dans δ, et $\Delta_{p,q} = \bot$ (faux) sinon.

3. Que représente la matrice Δ^n pour $n \in \mathbf{N}$? Comment obtenir la matrice $\Delta^{2^{n+1}}$ à partir de la matrice Δ^{2^n} ? En déduire que la fonction $f\colon n \mapsto 2^n$ préserve la régularité. *On pourra construire un automata dont l'ensemble des états est $Q \times \mathbf{B}^{Q^2}$.*

4. Toujours en utilisant la matrice Δ, montrer que la fonction $f\colon n \mapsto n^2$ préserve la régularité.

La suite de ce problème permet de donner une caractérisation de toutes les fonctions qui préservent la régularité. Un ensemble $U \subseteq \mathbb{N}$ est **ultimement périodique** s'il existe $p \geq 1$ tel que pour presque tous les entiers $n \in \mathbb{N}$, $n \in U$ si et seulement si $n + p \in U$: « presque tous les entiers » signifie ici « tous les entiers sauf un nombre fini » ; on pourra noter $\overset{\infty}{\forall}$ cette quantification. La famille des ensembles ultimement périodiques est la plus petite famille contenant les ensembles finis, les ensembles $[k]_m = \{k + mp \mid p \in \mathbb{N}\}$ pour tous $m \in \mathbb{N} \setminus \{0\}$ et $0 \leq k < m$, et qui est close par les opérations booléennes. On admet que l'ensemble $\lg(L) = \{|u| \mid u \in L\}$ des longueurs des mots d'un langage régulier L est ultimement périodique.

5. Montrer que si U est un ensemble ultimement périodique, alors $\{x \in \Sigma^* \mid |x| \in U\}$ est un langage régulier. En déduire que si $\Sigma = \{a\}$, un langage $L \subseteq \Sigma^*$ est régulier si et seulement si $\lg(L)$ est ultimement périodique.

Une fonction $f\colon \mathbb{N} \to \mathbb{N}$ est dite **ultimement périodique** s'il existe $p \geq 1$ tel que pour presque tous les entiers $n \in \mathbb{N}$, $f(n) = f(n+p)$. Une fonction $f\colon \mathbb{N} \to \mathbb{N}$ **préserve l'ultime périodicité** si $f^{-1}(U)$ est ultimement périodique, lorsque U l'est. On dit que la fonction f **est ultimement périodique modulo** m si la fonction $n \mapsto f(n) \mod m$ est ultimement périodique.

Pour $L \subseteq \Sigma^*$, on définit le langage

$$L'_f = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } v \in L\}$$

Finalement, introduisons quatre conditions dont on va montrer ensuite qu'elles sont équivalentes, fournissant un ensemble de caractérisations des fonctions préservant la régularité :

C1 f préserve la régularité ;

C2 pour tout langage régulier L, le langage L'_f est régulier ;

C3 f préserve l'ultime périodicité ;

C4 (i) f est ultimement périodique modulo m pour tout $m \geq 1$, et

(ii) $f^{-1}(\{n\})$ est ultimement périodique pour tout $n \in \mathbb{N}$.

6. Montrer que **C2** $\Rightarrow$ **C1**.

7. Montrer que **C3** $\Rightarrow$ **C2**.

8. Montrer que la condition **C4**(i) équivaut à ce que $f^{-1}([i]_m)$ soit ultimement périodique pour tout $m \geq 1$ et $0 \leq i \leq m-1$.

9. Montrer que **C1** $\Rightarrow$ **C4**(i). (*Indication : on pourra considérer le langage* $\left((a^m)^*a^k\right)_f$.)

10. Montrer que **C1** $\Rightarrow$ **C4**(ii). (*Indication : on pourra considérer le langage* $(a^*ba^n)_f \cap a^*b$.)

11. Toujours en utilisant le résultat de la question 8, montrer que **C4** $\Rightarrow$ **C3**.

12. Conclure et illustrer l'utilisation du théorème en considérant la fonction $f\colon n \mapsto \lfloor \log_2 n \rfloor$.

Les plus longs cycles et chemins

Un **graphe non orienté** G est un couple (V, E) où V est un ensemble de sommets et E un ensemble d'arêtes. Notons $n = |V|$, $m = |E|$. Les sommets u et v sont dits **voisins** s'il y a une arête notée $\{u, v\}$ entre u et v. Nous considérons les notations suivantes :

1. Le **degré** de v dans G, que l'on note $d_G(v)$, est le nombre de voisins de v.
2. Un **chemin (simple)** C allant de s à t est une suite $u_0, u_1, \ldots, u_\ell$ de sommets telle que $u_0 = s$, $u_\ell = t$, $(u_{i-1}, u_i) \in E$ pour tout $1 \leq i \leq \ell$ et telle que les sommets apparaissent une seule fois dans le chemin (sauf éventuellement ses extrémités). Un **cycle** est un chemin dont ses extrémités sont identiques ($u_0 = u_\ell$).
3. Un chemin (resp. cycle) est **hamiltonien** s'il traverse tous les sommets du graphe exactement une fois.

Question 1. Décrire un algorithme qui, étant donné un graphe G et une suite de sommets C, retourne un booléen qui indique si C est un cyle hamiltonien. Évaluer sa complexité.

Entrée : un graphe G et une suite de sommets $C = u_0, u_1, \ldots, u_\ell$
Sortie : un booléen b qui indique si C est hamiltonien.

1. Pour tout sommet v faire, $passer[v] = 0$;
2. si $u_0 \neq u_\ell$ ou si $\ell \neq n$ alors retourner faux ;
3. pour i allant de 0 à $\ell - 1$ faire
   (a) Si $\{u_i, u_{i+1}\} \notin E$ ou si $passer[u_{i+1}] > 0$ alors retourner faux ;
   (b) $passer[u_{i+1}] = 1$;
4. Retourner vrai ;

Complexité : Dans le pire des cas, on vérifie n fois qu'un couple de sommets est une arête du graphe. La complexité de l'algorithme est

1. $O(n^2)$ opérations si le graphe est codé avec une liste d'adjacence
2. $O(n)$ opérations si le graphe est codé avec une matrice d'adjacence

Question 2. Décrire un algorithme qui, étant donné un graphe G, retourne un booléen qui indique si le graphe G respecte la propriété suivante : la somme des degrés de toute paire de sommets u et v (avec $u \neq v$) non voisins vaut au moins n, c'est-à-dire $d_G(v) + d_G(u) > n$. Évaluer sa complexité.

Entrée : un graphe G
Sortie : un booléen b qui indique si le graphe G respecte la propriété.

1. pour tout sommet v, calculer le degré $d_G[v]$ de v.
2. pour tout sommet v, faire
   (a) pour tout sommet u, faire
      Si $\{v, u\} \notin E$ et $d_G[v] + d_G[u] \leq n$ alors retourner faux

3. Retourner vrai

Complexité : $O(n^2)$ opérations quel que soit le codage du graphe. Dans le pire des cas, l'instruction qui est la plus chère en terme d'opérations est le calcul des degrés.

Dans la suite, nous considérons un graphe non orienté de $n \geq 3$ sommets tel que la somme des degrés de toute paire de sommets non voisins est strictement supérieure à n.

Question 3. Soit v un sommet du graphe. Montrer que $n-1 \geq d_G(v) \geq 2$.

Le sommet v peut être voisin de tous les autres sommets sauf lui : $n-1 \geq d_G(v)$.
Nous avons $n-1 \geq d_G(v)$, $\forall v \in V$.
Nous allons montrer par contradiction que $d_G(v) \geq 2$.

1. Supposons que $d_G(v) = 0$. Cela signifie que tous les sommets de G ne sont pas voisin. Soit u un sommet de G. La relation $d_G(u)+d_G(v) > n$ implique que $d_G(u) \geq n$. Ceci est en contradiction avec le fait que $n-1 \geq d_G(u)$.
2. Supposons que $d_G(v) = 1$. Comme $n \geq 3$, il existe un sommet u du graphe non voisin de v. En utilisant les relations $d_G(v) = 1$, $d_G(u)+d_G(v) > n$ et $n-1 \geq d_G(u)$, on en déduit que $n-1 < n-1$ ce qui est une contradiction.

Question 4. Montrer que G est connexe.

Notons C_v la composante connexe de v.
Prouvons que G est connexe en montrant que $C_v = V$ par contradiction. Soit v et w deux sommets du graphe tel qu'il n'existe pas de chemin entre v et w. Par définition, $C_v \cap C_w = \emptyset$ et donc $n \geq |C_v| + |C_w|$.
Majorons le nombre de sommets dans C_v et C_w. Par définition du degré, nous avons $|C_v| \geq d_G(v) + 1$ et $|C_w| \geq d_G(w) + 1$.
Comme w et v ne sont pas voisins, nous avons $d_G(v) + d_G(w) > n$

$$|C_v| + |C_w| \geq d_G(v) + 1 + d_G(w) + 1 > n + 2$$

Cela contredit le fait que $n \geq |C_v| + |C_w|$.

Question 5. Soit $U = u_0, u_1, \ldots, u_{n-1}, u_0$ une suite de $n+1$ sommets contenant tous les sommets de G. Supposons que U n'est pas un cycle dans G.

Notons $a(U) = |\{i : \{u_i, u_{i+1}\} \in E \text{ et } 0 \leq i < n\}|$.

Question 5.1 Supposons que u_ℓ et $u_{\ell+1}$ ne sont pas voisins dans G. Montrer qu'il existe une suite $U' = u'_0, u'_1, \ldots, u'_{n-1}, u'_0$ de $n+1$ sommets contenant tous les sommets de G telle que $a(U') > a(U)$. *Indication : considérer l'ensemble des sommets*

$$X = \{u_{i+1} : i \neq \ell - 1 \wedge \{u_\ell, u_i\} \in E\}$$

Notons $X = \{u_{i+1} : i \neq \ell - 1 \wedge \{u_\ell, u_i\} \in E\}$. Par définition de X, nous avons $|X| \geq d_G(u_\ell) - 1$. Nous allons montrer par contradiction qu'il existe $u_{j+1} \in X$ voisin de $u_{\ell+1}$. Pour cela, nous supposons qu'il n'existe pas de tel sommet. Notons $B = (V \backslash \{u_\ell\}) \backslash X$. Par définition

$$|B| = n - 1 - |X| \leq n - 1 - (d_G(u_\ell) - 1) = n - d_G(u_\ell)$$

De plus, par hypothèse, tous les voisins de $u_{\ell+1}$ sont dans B donc $|B| \geq d_G(u_{\ell+1})$. En combinant les deux relations, nous obtenons

$$d_G(u_{\ell+1}) \leq |B| \leq n - d_G(u_\ell)$$

Ce qui implique

$$d_G(u_{\ell+1}) + d_G(u_\ell) \leq n$$

d'où une contradiction avec l'hypothèse $d_G(u_{\ell+1}) + d_G(u_\ell) > n$. Donc il existe $u_{j+1} \in X$ voisin de $u_{\ell+1}$.
Nous pouvons donc construire une suite U' de $n+1$ sommets contenant tous les sommets de G :

$$u_\ell, u_j, u_{j-1}, \ldots, u_{\ell+2}, u_{\ell+1}, u_{j+1}, u_{j+2}, \ldots u_{\ell-1}, u_\ell$$

telle que $a(U') = a(U) + 2 - \mathbb{1}_{\{u_j, u_{j+1}\} \in E} > a(U)$.

Question 5.2 Décrire un algorithme qui, étant donnée une suite $U = u_0, u_1, \ldots u_{n-1}, u_0$ contenant tous les sommets de G telle que $\{u_0, u_1\} \notin E$, retourne une suite $U' = u'_0, u'_1, \ldots u'_{n-1}, u'_0$ contenant tous les sommets de G telle que $a(U') > a(U)$. Évaluer sa complexité.

Entrée : un graphe G, une suite $U = u_0, u_1, \ldots u_{n-1}, u_0$ contenant tous les sommets de G telle que $\{u_0, u_1\} \notin E$
Sortie : une suite $U' = u'_0, u'_1, \ldots u'_{n-1}, u'_0$ telle que $a(U') > a(U)$

1. $j \leftarrow 2$
2. Tant que $\neg\ (((u_0, u_j) \in E) \wedge (u_1, u_{j+1}) \in E))$ faire
   exécution de la boucle au plus $n-1$ fois
   le test se fait en $O(1)$ opérations
   (a) $j \leftarrow j + 1$
3. Retourner $u_0, u_j, u_{j-1}, \ldots, u_2, u_1, u_{j+1}, u_{j+2}, \ldots u_{n-1}, u_0$ $O(n)$ opérations

Complexité : $O(n)$ opérations.

Question 6. Montrer que le graphe G admet un cycle hamiltonien.

Soit une suite $U = u_0, u_1, \ldots u_{n-1}, u_0$ contenant tous les sommets de G. Si U est un cycle, alors U est un cycle hamiltonien. Remarquons que $a(U) = n$
Nous prouvons ce résultat par contradiction. Supposons que G n'admet pas un cycle hamiltonien.

Soit $U = u_0, u_1, \dots u_{n-1}, u_0$ contenant tous les sommets de G telle que $a(U)$ soit la plus grande possible. Par la question 5.1, nous pouvons construire $U' = u'_0, u'_1, \dots u'_{n-1}, u'_0$ contenant tous les sommets de G telle que $a(U') > a(U)$. Ce contredit le fait que U est une suite contenant tous les sommets de G telle que $a(U)$ soit la plus grande possible.

Question 7. Décrire un algorithme qui, étant donné un graphe G, retourne un cycle hamiltonien. Évaluer sa complexité.

Entrée : un graphe G,
Sortie : un cycle C

1. Construire une suite de sommets arbitraire $U = u_0, u_1, \dots u_{n-1}, u_0$
   $O(n)$ opérations
2. Tant que $a(U) < n$ faire au pire des cas, la boucle est exécutée n fois
   (a) $\ell \leftarrow 0$ $O(1)$ opérations
   (b) tant que $\{u_\ell, u_\ell\} \in E$ faire $\ell \leftarrow \ell + 1$
   (c) $U \leftarrow$ Algorithme de la question 5.2 prenant en entrée $u_\ell, u_{\ell+1}, \dots, u_{n-1}, u_0, \dots, u_{\ell-1}, u_\ell$ $O(n)$ opérations
3. Retourner U

Complexité : $O(n^2)$ opérations si le graphe est codé sous forme de matrice d'adjacence.

Un **graphe orienté** G est un couple (V, A) où V est un ensemble de sommets et $A \subseteq V \times V$ est un ensemble d'arcs. Rappelons les notations dans le contexte des graphes orientés :

1. L'arc de u vers v que l'on notera par $(u \to v)$ est un arc **entrant** dans v.
2. Un chemin C allant de s à t est une suite $u_0, u_1, \dots, u_\ell$ de sommets telle que $u_0 = s$, $u_\ell = t$, $(u_{i-1} \to u_i) \in A$ pour tout $1 \leq i \leq \ell$. Un **cycle** est un chemin dont ses extrémités sont identiques ($u_0 = u_\ell$).

Nous considérons un graphe orienté sans cycle sans contrainte sur les degrés. Soit G un graphe orienté sans cycle.

Question 8. Montrer que G possède au moins un sommet u_0 sans arc entrant.

Prouvons le par contradiction. Supposons que chaque sommet v de G possède un arc entrant. Nous allons considérer un chemin $P = u_0, u_1, \dots, u_\ell$ tel que
— pour tout $0 \leq i < \ell$, $(u_i \to u_{i+1})$ est un arc de G.
— le chemin soit le plus long possible.
Comme G est sans cycle, tous les sommets sont distincts. Comme u_0 possède un arc entrant $(v \to u_0)$, v n'apparait pas dans P (sinon G admettrait un cycle), donc on peut construire $P' = u_0, u_1, \dots, u_\ell, u_{\ell+1}$ plus long que P. Ceci amène à une contradiction.

Question 9. Décrire un algorithme qui, étant donné un graphe orienté sans cycle G, retourne un chemin hamiltonien (s'il existe). Évaluer sa complexité.

Observons qu'un chemin hamiltonien $C = u_0, \ldots, u_{n-1}$ passe par tous les sommets du graphe. Comme le graphe est sans cycle, il suffit de noter qu'il n'existe pas d'arc $(u_j \to u_i)$ avec $j > i$.

Cela signifie donc que u_0 est un sommet ne possédant pas d'arc entrant dans G, u_1 est un sommet ne possédant pas d'arc entrant dans G privé du sommet u_0, *etc.*

L'idée est de calculer tous les sommets ne possédant pas d'arc entrant dans G.

Entrée : un graphe G,
Sortie : un chemin P s'il en existe sinon $\emptyset$

1. Trouver un sommet u_0 ne possédant pas d'arc entrant dans G ;
   au plus $O(|V|^2)$ opérations
2. $P \leftarrow u_0$
3. pour i allant 1 à $n-1$ faire exécution de la boucle au plus $n-1$ fois
   (a) Construire $G_i = (V \backslash P, A \backslash (V \times P \cup P \times V))$
   au total sur toute l'exécution, au plus $O(|A|)$ opérations
   (b) Trouver un sommet u_i ne possédant pas d'arc entrant dans G_i ;
   au plus $O(|V|^2)$ opérations
   (c) S'il existe un arc $(u_{i-1} \to u_i)$ dans G $O(|1|)$ opérations
      i. alors $P \leftarrow P \cup \{u_i\}$
      ii. sinon retourner $\emptyset$
4. Retourner P

Complexité : $O(|V|^3 + |A|)$ opérations. En utilisant un tableau des degrés entrants, la complexité de l'algorithme est $O(|V|^2 + |A|)$. Avant la boucle pour, nous pouvons construire un tableau T tel que $T[u]$ correspond au degré entrant de u de G. Ensuite, il faut réactualiser le tableau T pour que $T[u]$ corresponde au degré entrant de u de G_i. Pour cela, à la place de l'instruction (3.a), il faut diminuer la valeur $T[u]$ s'il existe un arc $(u_{i-1} \to u)$ dans G. Au total, cela nécessite au plus $O(|V|^2)$ opérations. Ensuite, l'instruction (3.a) correspond à trouver le minimum dans un tableau.

Un chemin C de s à t est dit **un plus long chemin** s'il a le plus grand nombre d'arcs parmi tous les chemins allant de s à t.

Question 10. Soit $C = u_0, u_1, \ldots, u_\ell$ un plus long chemin allant de u_0 à u_ℓ dans un graphe orienté sans cycle. Montrer que les sous-chemins de C allant de u_0 à u_i (avec $0 \leq i \leq \ell$) et allant de u_i à u_ℓ sont aussi des plus longs chemins. Est-ce le cas dans un graphe orienté avec cycle ?

Prouvons le résultat par contradiction. Supposons que le sous-chemin C_1 allant de u_0 à u_i de C n'est pas un plus long chemin. Soit C' un plus long chemin. allant de u_0 à u_i. Nous allons construire un chemin $C^* = C', u_i, \ldots, u_\ell$.

Comme le graphe n'a pas de cycle, aucun sommet de C' n'est dans le chemin $u_i, \ldots, u_\ell$. Comme $|C'| > |C_1|$, alors $|C^*| > |C|$. Ce qui est en contradiction avec le fait que C est un plus long chemin allant de u_0 à u_ℓ.

Ce n'est pas le cas pour un graphe orienté avec cycle : il suffit de prendre un cycle orienté où on peut aller dans les deux sens.

Question 11. Donner la formule de récurrence qui permet de calculer la longueur d'un chemin le plus long finissant par u.

Notons $long(u)$ la longueur du chemin le plus long finissant par u.

S'il n'existe pas de chemin finissant par u (c'est-à-dire le degré entrant de u est égal à 0), alors par convention on supposera que $long(u) = 0$.

Soit $L = u_1, \ldots, u_{\ell-1}, u_\ell$ un chemin le plus long finissant par u_ℓ. Par la question précédente, le chemin $u_1, \ldots, u_{\ell-1}$ est un chemin le plus long allant u_1 de $u_{\ell-1}$. Donc la longueur du plus long chemin allant u_1 de u_ℓ est égal à

$$1+ \text{ la longueur du plus long chemin allant de } u_1 \text{ à } u_{\ell-1}$$

Donc $long(u_\ell) = 1 + long(u_{\ell-1})$.

Comme un chemin finissant par u_ℓ doit passer par un sommet de $\Gamma^-(v)$ avec $\Gamma^-(v) = \{u : (u \to v) \in A\}$, la longueur du plus long chemin finissant par u_ℓ est égal à 1+ la longueur du plus long chemin finissant par v avec $v \in \Gamma^-(u_\ell)$. Donc

$$long(u_\ell) = \begin{cases} 1 + \max\{long(v) : v \in \Gamma^-(u_\ell)\} & \text{si} \quad u_\ell \text{ a un degré entrant} \\ 0 & \text{si} \quad \ell = 1 \end{cases}$$

Question 12. Donner un algorithme qui retourne la longueur d'un chemin le plus long finissant par u, pour tout sommet u du graphe. Évaluer la complexité.

Soit $long : V \to \mathbb{N}$ un tableau d'entiers tel que $long(u)$ désigne la longueur d'un plus long chemin finissant par u.

Entrée : un graphe orienté sans cycle G
Sortie : un tableau entier $long : V \to \mathbb{N}$

1. pour tout sommet v faire,
   (a) $d^-(v) \leftarrow$ le degré entrant de v ; — cela dépend du codage
   (b) $long(v) \leftarrow 0$; — $O(n)$ opérations au total
2. trier les sommets en fonction de d^- croissant ;
3. $S \leftarrow V$;
4. Tant que $(S \neq \emptyset)$ faire — la boucle est exécutée $O(n)$ fois
   (a) choisir v dans S tel que $d^-(v) = 0$

(b) extraire v de S

(c) pour tout t tel qu'il existe un arc de v vers t faire — la boucle est exécutée $O(m)$ fois

i. si ($long(v) + 1 > long(t)$) alors $long(t) \leftarrow long(v) + 1$;

ii. $d^-(t) = d^-(t) - 1$

5. retourner le tableau $long$;

Complexité : $O(n^2)$ opérations.

De plus pour éviter de gérer le tableau des degrés sortants, nous pouvons renuméroter les sommets tel que s'il existe un arc $(i \to j)$, alors $i < j$ et traiter les sommets dans cet ordre. Pour cela, il suffit de faire un tri topologique ($O(|A|^2)$ opérations).

Question 13. Adapter l'algorithme précédent pour qu'il retourne le plus long chemin dans le graphe. Évaluer la complexité.

Entrée : un graphe orienté sans cycle G
Sortie : un chemin π

1. pour tout sommet v faire,
   (a) $d^-(v) \leftarrow$ le degré entrant de v ; — cela dépend du codage
   (b) $long(v) \leftarrow 0$; — $O(n)$ opérations au total
   (c) $\pi(v) \leftarrow \{v\}$; — $O(n)$ opérations au total
2. trier les sommets en fonction de d^- croissant ;
3. $S \leftarrow V$;
4. Tant que ($S \neq \emptyset$) faire — la boucle est exécutée $O(n)$ fois
   (a) choisir v dans S tel que $d^-(v) = 0$
   (b) extraire v de S
   (c) pour tout t tel qu'il existe un arc de v vers t faire — la boucle est exécutée $O(m)$ fois
      i. si ($long(v) + 1 > long(t)$) alors $long(t) \leftarrow long(v) + 1$;
      ii. $d^-(t) = d^-(t) - 1$
      iii. $\pi(t) = t, \pi(v)$
5. trouver un sommet v tel que $long(v) = \max\{long(u) : u \in V\}$
6. retourner $\pi(v)$;

Complexité : $O(n^2)$ opérations.

Préserver la régularité coûte que coûte - Correction

Soit Σ un alphabet fini et non vide. La longueur d'un mot $w \in \Sigma^*$ est notée $|w|$. Un **automate fini** $\mathcal{A}$ sur l'alphabet Σ est la donnée d'un tuple $\mathcal{A} = \langle Q, \delta, I, T\rangle$ où Q est un ensemble fini d'états, $\delta \subseteq Q \times \Sigma \times Q$ est l'ensemble des transitions, $I \subseteq Q$ est l'ensemble des états initiaux et $T \subseteq Q$ est l'ensemble des états terminaux. La transition $(p, \sigma, q) \in \delta$ est notée par $p \xrightarrow{\sigma} q$. Un **calcul** c de $\mathcal{A}$ est une séquence finie de transitions qui forment un chemin dans le graphe de $\mathcal{A}$, $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$: on note un tel calcul $c = q_0 \xrightarrow{\sigma_1\sigma_2\cdots\sigma_n} q_n$. L'**étiquette** de c est le mot $\sigma_1\sigma_2\cdots\sigma_n$ de Σ^*. Le calcul est **acceptant** si $q_0 \in I$ et $q_n \in T$. Le **langage** de $\mathcal{A}$ est le sous-ensemble $\mathcal{L}(\mathcal{A})$ de Σ^* qui contient l'ensemble des étiquettes des calculs acceptants de $\mathcal{A}$. Un tel langage est dit **régulier**. L'automate $\mathcal{A}$ est dit **déterministe** si I possède un unique état et pour tout $p \in Q$ et $\sigma \in \Sigma$, il existe au plus un état $q \in Q$ tel que $(p, \sigma, q) \in \delta$.

Ce sujet propose de caractériser les applications $f\colon \mathbf{N} \to \mathbf{N}$ telle que pour tout langage régulier, le langage $L_f = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } uv \in L\}$ est aussi régulier. On dit qu'une telle fonction **préserve la régularité**.

1. Montrer que la fonction identité préserve la régularité, c'est-à-dire que si L est un langage régulier, alors $L' = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |u| = |v| \text{ et } uv \in L\}$ est régulier.

Solution : Soit $\mathcal{A} = \langle Q, \delta, I, T\rangle$ un automate reconnaissant le langage L. On construit un automate $\mathcal{A}' = \langle Q', \delta', I', T'\rangle$ reconnaissant L' : l'idée est de deviner un état $\widetilde{q}$ qui peut être atteint après avoir lu le mot u dans $\mathcal{A}$ et qui permet d'accepter un mot v de même longueur que u. Pour cela, on pose $Q' = Q^3$ (le premier état permet de simuler $\mathcal{A}$ sur le mot u, le second état est $\widetilde{q}$, le troisième état permet de simuler une exécution acceptante depuis $\widetilde{q}$ avec un mot v de même longueur que u), $I' = I \times \{(\widetilde{q}, \widetilde{q}) \mid \widetilde{q} \in Q\}$, $T' = \{(\widetilde{q}, \widetilde{q}, q) \mid \widetilde{q} \in Q, q \in T\}$ et

$$\delta' = \{\big((p, \widetilde{q}, p'), \sigma, (q, \widetilde{q}, q')\big) \mid \widetilde{q} \in Q,\ (p, \sigma, q) \in \delta,\ \exists \sigma' \in \Sigma \quad (p', \sigma', q') \in \delta\}$$

On peut montrer par récurrence sur la longueur de $u = \sigma_1\sigma_2\cdots\sigma_n$ que les calculs de $\mathcal{A}'$ sur u sont ceux de la forme

$$(q_0, \widetilde{q}, \widetilde{q}) \xrightarrow{\sigma_1} (q_1, \widetilde{q}, q'_1) \xrightarrow{\sigma_2} (q_2, \widetilde{q}, q'_2) \cdots \xrightarrow{\sigma_n} (q_n, \widetilde{q}, q'_n)$$

tels que $q_0 \in I$ et $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$ dans $\mathcal{A}$ et il existe un mot $v = \sigma'_1\sigma'_2\cdots\sigma'_n$ de même longueur que u tel que $\widetilde{q} \xrightarrow{\sigma'_1} q'_1 \xrightarrow{\sigma'_2} q'_2 \cdots \xrightarrow{\sigma'_n} q'_n$ dans $\mathcal{A}$. Un tel calcul est acceptant si et seulement si $q_n = \widetilde{q}$ et $q'_n \in T$, c'est-à-dire si et seulement la concaténation des calculs précédents

$$q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n \xrightarrow{\sigma'_1} q'_1 \xrightarrow{\sigma'_2} q'_2 \cdots \xrightarrow{\sigma'_n} q'_n$$

sur le mot uv est acceptant dans $\mathcal{A}$. Cela prouve que $\mathcal{A}'$ reconnait le langage L'.

2. Montrer que toute fonction affine, de la forme $f : n \mapsto an + b$ avec $a, b \in \mathbf{N}$, préserve la régularité. *On pourra commencer par considérer les cas particuliers où $a = 0$, puis $b = 0$.*

Solution : Montrons tout d'abord que toute fonction constante $f_b : n \mapsto b$ (avec $b \in \mathbb{N}$) préserve la régularité. Soit $\mathcal{A} = \langle Q, \delta, I, T \rangle$ un automate reconnaissant le langage L. On construit un automate $\mathcal{A}' = \langle Q, \delta, I, T' \rangle$ reconnaissant L_f. Seuls les états terminaux sont différents, puisqu'une fois lu le mot u, il faut vérifier s'il est possible d'atteindre un état terminal en exactement b étapes. On peut donc poser $T' = \{q \in Q \mid \exists v \in \Sigma^b, \exists q' \in T \quad q \xrightarrow{v} q'\}$. Un tel ensemble est calculable puisque Σ^b est un langage fini.
Ensuite, il est facile de modifier la construction de la question précédente pour une fonction linéaire $f_a : n \mapsto an$ avec $a \geq 1$. En effet, il suffit de réaliser a étapes de calculs dans v, pour chaque lettre de u. On peut donc poser

$$\delta' = \{\big((p, \widetilde{q}, p'), \sigma, (q, \widetilde{q}, q')\big) \mid \widetilde{q} \in Q,\ (p, \sigma, q) \in \delta,\ \exists v \in \Sigma^a \quad p' \xrightarrow{v} q' \text{ dans } \mathcal{A}\}$$

Comme précédemment, ce nouvel ensemble de transition est calculable puisque Σ^a est un ensemble fini.
Considérons alors une fonction affine $f : n \mapsto an + b$. Si $a = 0$, c'est une fonction constante dont on sait déjà qu'elle préserve la régularité. Sinon, $f = f_a + f_b$. Or,

$$\begin{aligned}
(L_{f_b})_{f_a} &= \{u \in \Sigma^* \mid \exists v_a \in \Sigma^* \quad |v_a| = f_a(|u|) \text{ et } uv_a \in L_{f_b}\} \\
&= \{u \in \Sigma^* \mid \exists v_a, v_b \in \Sigma^* \quad |v_a| = f_a(|u|),\ |v_b| = f_b(|uv_a|) \text{ et } uv_av_b \in L\} \\
&= \{u \in \Sigma^* \mid \exists v_a, v_b \in \Sigma^* \quad |v_av_b| = a|u| + b = f(|u|) \text{ et } uv_av_b \in L\} \\
&= L_f
\end{aligned}$$

Puisque f_a et f_b préservent la régularité, $(L_{f_b})_{f_a} = L_f$ est un langage régulier si L l'est. Donc toute fonction affine préserve la régularité.

Pour généraliser davantage l'étude de la préservation de la régularité, introduisons la notion de **matrice de transitions** d'un automate $\mathcal{A} = \langle Q, \delta, I, T \rangle$: il s'agit de la matrice carrée booléenne $\Delta = (\Delta_{p,q})_{p,q \in Q} \in \mathbf{B}^{Q^2}$ indexée par les états de Q, avec $\mathbf{B} = \{\top, \bot\}$, tel que $\Delta_{p,q} = \top$ (vrai) s'il existe un symbole $\sigma \in \Sigma$ avec une transition $p \xrightarrow{\sigma} q$ dans δ, et $\Delta_{p,q} = \bot$ (faux) sinon.

3. Que représente la matrice Δ^n pour $n \in \mathbf{N}$? Comment obtenir la matrice $\Delta^{2^{n+1}}$ à partir de la matrice Δ^{2^n} ? En déduire que la fonction $f : n \mapsto 2^n$ préserve la régularité. *On pourra construire un automata dont l'ensemble des états est $Q \times \mathbf{B}^{Q^2}$.*

Solution : La matrice Δ est la matrice d'adjacence du graphe de l'automate $\mathcal{A}$. La matrice Δ^n représente donc la matrice des chemins de longueur n dans ce graphe, c'est-à-dire que $(\Delta^n)_{p,q} = \top$ si et seulement s'il existe un calcul de longueur n allant de p à q.
On remarque que $\Delta^{2^{n+1}} = \left(\Delta^{2^n}\right)^2$. Utilisons cela pour prouver que la fonction $f : n \mapsto 2^n$ préserve la régularité. Pour cela, considérons un langage L reconnu par l'automate $\mathcal{A} = \langle Q, \delta, I, T \rangle$. Soit $\mathcal{A}' = \langle Q', \delta', I', T' \rangle$ tel que $Q' = Q \times \mathbf{B}^{Q^2}$, $I' = I \times \{\Delta\}$, $T' = \{(q, M) \mid \exists q' \in T \quad M_{q,q'} = \top\}$ et $\delta' = \{\big((p, M), \sigma, (q, M^2)\big) \mid (p, \sigma, q) \in \delta\}$.

On peut montrer par récurrence sur la longueur n du mot u que les calculs de $\mathcal{A}'$ sur $u = \sigma_1\sigma_2\cdots\sigma_n$ sont de la forme

$$(q_0, \Delta) \xrightarrow{\sigma_1} (q_1, \Delta^2) \xrightarrow{\sigma_2} (q_2, \Delta^{2^2}) \cdots \xrightarrow{\sigma_n} (q_n, \Delta^{2^n})$$

avec $q_0 \in I$. Ce calcul est acceptant si et seulement s'il existe un état $q \in T$ tel que $\left(\Delta^{2^n}\right)_{q_n,q} = \top$, c'est-à-dire s'il existe un mot v de longueur 2^n et un calcul $q_n \xrightarrow{v} q$. Le calcul de $\mathcal{A}$

$$q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n \xrightarrow{v} q$$

est donc un calcul acceptant d'étiquette uv. Ainsi, $\mathcal{A}'$ reconnait le langage L_f.

4. Toujours en utilisant la matrice Δ, montrer que la fonction $f\colon n \mapsto n^2$ préserve la régularité.

Solution : On s'inspire de la méthode utilisée à la question précédente. On remarque que $(n+1)^2 = n^2+2n+1$, donc $\Delta^{(n+1)^2} = \Delta^{n^2}\Delta^{2n+1}$. Il faut donc non seulement maintenir Δ^{n^2} dans l'état, mais aussi Δ^{2n+1}, qu'on peut également calculer grâce à la relation $\Delta^{2(n+1)+1} = \Delta^{2n+1}\Delta^2$. Si L est reconnu par l'automate $\mathcal{A} = \langle Q, \delta, I, T\rangle$, le langage L_f est donc reconnu par l'automate $\mathcal{A}' = \langle Q', \delta', I', T'\rangle$ tel que $Q' = Q \times \left(\mathbf{B}^{Q^2}\right)^2$, $I' = I \times \{(Id, \Delta)\}$ avec Id la matrice identité ($\top$ sur la diagonale et $\bot$ ailleurs), $T' = \{(q, M, N) \mid \exists q' \in T \quad M_{q,q'} = \top\}$ et $\delta' = \{\big((p, M, N), \sigma, (q, MN, N\Delta^2)\big) \mid (p, \sigma, q) \in \delta\}$.

La suite de ce problème permet de donner une caractérisation de toutes les fonctions qui préservent la régularité. Un ensemble $U \subseteq \mathbb{N}$ est **ultimement périodique** s'il existe $p \geq 1$ tel que pour presque tous les entiers $n \in \mathbb{N}$, $n \in U$ si et seulement si $n + p \in U$: « presque tous les entiers » signifie ici « tous les entiers sauf un nombre fini » ; on pourra noter $\overset{\infty}{\forall}$ cette quantification. La famille des ensembles ultimement périodiques est la plus petite famille contenant les ensembles finis, les ensembles $[k]_m = \{k + mp \mid p \in \mathbb{N}\}$ pour tous $m \in \mathbb{N} \setminus \{0\}$ et $0 \leq k < m$, et qui est close par les opérations booléennes. On admet que l'ensemble $\lg(L) = \{|u| \mid u \in L\}$ des longueurs des mots d'un langage régulier L est ultimement périodique.

5. Montrer que si U est un ensemble ultimement périodique, alors $\{x \in \Sigma^* \mid |x| \in U\}$ est un langage régulier. En déduire que si $\Sigma = \{a\}$, un langage $L \subseteq \Sigma^*$ est régulier si et seulement si $\lg(L)$ est ultimement périodique.

Solution : Tout ensemble ultimement périodique U peut s'écrire comme $V \cup W$ avec V un ensemble fini et $W = \{\alpha + kp \mid k \in \mathbb{N}\}$ pour un certain entier α (à partir duquel U est périodique). Le langage $\{x \in \Sigma^* \mid |x| \in W\}$ est reconnu par un automate fini sous forme d'un lasso (ou d'une poêle à frire...) commençant par une suite de α transitions étiquetées par Σ, puis d'un cycle de longueur p où chaque transition est aussi étiquetée par Σ. De plus, $\{x \in \Sigma^* \mid |x| \in V\}$ est un langage fini et donc régulier. Puisque les langages réguliers sont clos par union, $\{x \in \Sigma^* \mid |x| \in U\}$ est régulier.
Dans le cas où $\Sigma = \{a\}$, on a $L = \{x \in \Sigma^* \mid |x| \in \lg(L)\}$. Donc si $\lg(L)$ est ultimement périodique, alors L est régulier. La réciproque est admise dans l'énoncé.

Une fonction $f\colon \mathbb{N} \to \mathbb{N}$ est dite **ultimement périodique** s'il existe $p \geq 1$ tel que pour presque tous les entiers $n \in \mathbb{N}$, $f(n) = f(n + p)$. Une fonction $f\colon \mathbb{N} \to \mathbb{N}$ **préserve l'ultime périodicité** si $f^{-1}(U)$ est ultimement périodique, lorsque U l'est. On dit que la fonction f

est ultimement périodique modulo m si la fonction $n \mapsto f(n) \mod m$ est ultimement périodique.

Pour $L \subseteq \Sigma^*$, on définit le langage

$$L'_f = \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } v \in L\}$$

Finalement, introduisons quatre conditions dont on va montrer ensuite qu'elles sont équivalentes, fournissant un ensemble de caractérisations des fonctions préservant la régularité :

C1 f préserve la régularité ;

C2 pour tout langage régulier L, le langage L'_f est régulier ;

C3 f préserve l'ultime périodicité ;

C4 (i) f est ultimement périodique modulo m pour tout $m \geq 1$, et

(ii) $f^{-1}(\{n\})$ est ultimement périodique pour tout $n \in \mathbb{N}$.

6. Montrer que **C2** $\Rightarrow$ **C1**.

Solution : Soit f satisfaisant **C2**. Montrons que f préserve la régularité. Pour cela, soit L un langage régulier, reconnu par un automate déterministe $\mathcal{A} = \langle Q, \delta, \{q_0\}, T\rangle$. Pour $q \in Q$ et $F \subseteq Q$, on note $\mathcal{A}_q^F$ l'automate $\langle Q, \delta, \{q\}, F\rangle$. Alors,

$$\begin{aligned}
L_f &= \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } uv \in L\} \\
&= \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } \exists q \in Q, q_f \in T \quad q_0 \xrightarrow{u} q \xrightarrow{v} q_f\} \\
&= \bigcup_{q \in Q} \{u \in \Sigma^* \mid q_0 \xrightarrow{u} q\} \cap \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } v \in \mathcal{L}(\mathcal{A}_q^T)\} \\
&= \bigcup_{q \in Q} \mathcal{L}(\mathcal{A}_{q_0}^{\{q\}}) \cap \mathcal{L}(\mathcal{A}_q^T)'_f
\end{aligned}$$

Pour tout q, $\mathcal{L}(\mathcal{A}_{q_0}^{\{q\}})$ et $\mathcal{L}(\mathcal{A}_q^T)$ sont des langages réguliers, et par **C2**, $\mathcal{L}(\mathcal{A}_q^T)'_f$ aussi. Puisque l'ensemble des langages réguliers est clos par union et intersection, le langage L_f est régulier et f préserve donc la régularité.

7. Montrer que **C3** $\Rightarrow$ **C2**.

Solution : Soit f satisfaisant **C3**. Considérons un langage régulier L. Notons que

$$\begin{aligned}
L'_f &= \{u \in \Sigma^* \mid \exists v \in \Sigma^* \quad |v| = f(|u|) \text{ et } v \in L\} \\
&= \{u \in \Sigma^* \mid \exists n \in \lg(L) \quad n = f(|u|)\} \\
&= \{u \in \Sigma^* \mid f(|u|) \in \lg(L)\} \\
&= \{u \in \Sigma^* \mid |u| \in f^{-1}(\lg(L))\}
\end{aligned}$$

Puisque L est régulier, $\lg(L)$ est ultimement périodique (propriété admise par l'énoncé). Par **C3**, $f^{-1}(\lg(L))$ est donc ultimement périodique aussi. Par la question 5, L'_f est donc un langage régulier.

8. Montrer que la condition **C4**(i) équivaut à ce que $f^{-1}([i]_m)$ soit ultimement périodique pour tout $m \geq 1$ et $0 \leq i \leq m-1$.

Solution : Soit $m \geq 1$. On abrège « ultimement périodique » en « u.p. ». On a les équivalences suivantes :

$$
\begin{aligned}
& f^{-1}([i]_m) \text{ est u.p. pour tout } 0 \leq i \leq m-1 \\
\iff & \bigwedge_{i=0}^{m-1} \exists p_i \geq 1 \quad f^{-1}([i]_m) \text{ est u.p. de période } p_i \\
\iff & \exists p \geq 1 \bigwedge_{i=0}^{m-1} f^{-1}([i]_m) \text{ est u.p. de période } p \qquad (p = \mathrm{ppcm}_i p_i) \\
\iff & \exists p \geq 1 \bigwedge_{i=0}^{m-1} \overset{\infty}{\forall} n \quad \Big(n \in f^{-1}([i]_m) \Leftrightarrow n+p \in f^{-1}([i]_m)\Big) \\
\iff & \exists p \geq 1 \bigwedge_{i=0}^{m-1} \overset{\infty}{\forall} n \quad \Big(f(n) \in [i]_m \Leftrightarrow f(n+p) \in [i]_m\Big) \\
\iff & \exists p \geq 1 \overset{\infty}{\forall} n \bigwedge_{i=0}^{m-1} \Big(f(n) \in [i]_m \Leftrightarrow f(n+p) \in [i]_m\Big) \\
\iff & \exists p \geq 1 \overset{\infty}{\forall} n \quad f(n) = f(n+p) \mod m \\
\iff & f \text{ est ultimement périodique modulo } m
\end{aligned}
$$

9. Montrer que **C1** $\Rightarrow$ **C4**(i). (*Indication : on pourra considérer le langage* $\left((a^m)^* a^k\right)_f$.)

Solution : Soit f satisfaisant **C1**. Soit $m \geq 1$. Pour $k \in \{0, 1, \ldots, m-1\}$, considérons le langage régulier $(a^m)^* a^k$. On a

$$
\begin{aligned}
\left((a^m)^* a^k\right)_f &= \{u \mid \exists v \quad |v| = f(|u|) \text{ et } uv \in \{a^{mn+k} \mid n \in \mathbb{N}\} \\
&= \{a^i \mid a^{i+f(i)} \in \{a^{mn+k} \mid n \in \mathbb{N}\} \\
&= \{a^i \mid i + f(i) \equiv k \mod m\}
\end{aligned}
$$

Par **C1**, ce langage est régulier, donc l'ensemble de ses longueurs est ultimement périodique. Or,

$$\mathrm{lg}\left(\left((a^m)^* a^k\right)_f\right) = \{i \mid i + f(i) \equiv k \mod m\} = g^{-1}([k]_m)$$

avec $g\colon n \mapsto n + f(n)$. Puisque c'est vrai pour tout $k \in \{0, 1, \ldots, m-1\}$, par le début de la question, g est donc ultimement périodique modulo m. Puisque la fonction $n \mapsto (-n) \mod m$ est aussi ultimement périodique modulo m, leur somme l'est aussi : la fonction f est donc ultimement périodique modulo m (pour tout m).

10. Montrer que **C1** $\Rightarrow$ **C4**(ii). (*Indication : on pourra considérer le langage* $(a^* b a^n)_f \cap a^* b$.)

Solution : Soit f satisfaisant **C1**. Soit $n \in \mathbb{N}$. Considérons le langage régulier a^*ba^n. On a

$$\begin{aligned}(a^*ba^n)_f \cap a^*b &= \{a^mb \mid \exists v \quad |v| = f(|a^mb|) \text{ et } a^mbv \in \{a^mba^n \mid m \in \mathbb{N}\}\} \\ &= \{a^mb \mid \exists v \quad |v| = f(m+1) \text{ et } v = a^n\} \\ &= \{a^mb \mid n = f(m+1)\} = \{a^mb \mid m+1 \in f^{-1}(\{n\})\}\end{aligned}$$

Par **C1** et clôture de l'ensemble des langages réguliers par intersection, ce langage est régulier. Ainsi, l'ensemble de ses longueurs est ultimement périodique :

$$\lg(\{a^mb \mid m+1 \in f^{-1}(\{n\})\}) = \{m+1 \mid m+1 \in f^{-1}(\{n\})\} = f^{-1}(\{n\}) \setminus \{0\}$$

L'ensemble $f^{-1}(\{n\})$ est donc également ultimement périodique.

11. Toujours en utilisant le résultat de la question 8, montrer que **C4** $\Rightarrow$ **C3**.

Solution : Soit f satisfaisant **C4**. Soit U un ensemble ultimement périodique de période p. Montrons que $f^{-1}(U)$ est ultimement périodique. L'ensemble U peut s'écrire

$$U = (F \cup [i_1]_p \cup [i_2]_p \cup \cdots \cup [i_k]_p) \setminus G$$

avec F et G des ensembles finis et $0 \leq i_1 < i_2 < \cdots < i_k \leq p-1$. Ainsi,

$$\begin{aligned}f^{-1}(U) &= f^{-1}\big((F \cup [i_1]_p \cup [i_2]_p \cup \cdots \cup [i_k]_p) \setminus G\big) \\ &= \big(f^{-1}(F) \cup f^{-1}([i_1]_p) \cup f^{-1}([i_2]_p) \cup \cdots \cup f^{-1}([i_k]_p)\big) \setminus f^{-1}(G) \\ &= \left(\bigcup_{n\in F} f^{-1}(\{n\}) \cup f^{-1}([i_1]_p) \cup f^{-1}([i_2]_p) \cup \cdots \cup f^{-1}([i_k]_p)\right) \setminus \bigcup_{n\in G} f^{-1}(\{n\})\end{aligned}$$

Par **C4**, la question 8 et la clôture des ensembles ultimement périodiques par opérations booléennes, cet ensemble est ultimement périodique.

12. Conclure et illustrer l'utilisation du théorème en considérant la fonction $f\colon n \mapsto \lfloor \log_2 n \rfloor$.

Solution : On a donc montré les chaînes d'implication **C2** $\Rightarrow$ **C1** $\Rightarrow$ **C4** $\Rightarrow$ **C3** $\Rightarrow$ **C2** donc les quatre propriétés sont équivalentes. Montrons que la fonction $f\colon n \mapsto \lfloor \log_2 n \rfloor$ ne satisfait pas **C4**(i) pour conclure qu'elle ne préserve pas la régularité. Si f était ultimement périodique modulo 2, alors $g^{-1}(\{1\})$ serait ultimement périodique, avec $g\colon n \mapsto f(n) \bmod 2$. Or

$$\begin{aligned}g^{-1}(\{1\}) &= \{n \mid \lfloor \log_2 n \rfloor \equiv 1 \mod 2\} = \{n \mid \exists k \in \mathbb{N} \quad \lfloor \log_2 n \rfloor = 2k+1\} \\ &= \{n \mid \exists k \in \mathbb{N} \quad 2k+1 \leq \log_2 n < 2k+2\} \\ &= \{n \mid \exists k \in \mathbb{N} \quad 2^{2k+1} \leq n < 2^{2k+2}\}\end{aligned}$$

Pour tout $k \in \mathbb{N}$, l'ensemble $g^{-1}(\{1\})$ contient 2^{2k+1} et 2^{2k+3}, mais aucun entier dans l'intervalle $\{2^{2k+2}, 2^{2k+2}+1, \ldots, 2^{2k+3}-1\}$ de longueur $2^{2k+3} - 2^{2k+2} = 2^{2k+2}$. Ainsi, pour k assez grand, $g^{-1}(\{1\})$ ne peut pas avoir une période inférieure à 2^{2k+2}. L'ensemble $g^{-1}(\{1\})$ ne peut donc avoir aucune période, et n'est donc pas ultimement périodique.

Inspiré par On Regularity-Preserving Functions *de D. Kozen, Cornell University, 1995.*