Jeux sur des arbres (*tries on tries*).

Épreuve pratique d'algorithmique et de programmation
Concours commun des Écoles normales supérieures

Durée de l'épreuve : 3 heures 30 minutes

Juin/Juillet 2021

ATTENTION !

N'oubliez en aucun cas de recopier votre u_0
à l'emplacement prévu sur votre fiche réponse

Important.

Il vous a été donné un numéro u_0 qui servira d'entrée à vos programmes. Les réponses attendues sont généralement courtes et doivent être données sur la fiche réponse fournie à la fin du sujet. À la fin du sujet, vous trouverez en fait deux fiches réponses. La première est un exemple des réponses attendues pour un $\widetilde{u_0}$ particulier (précisé sur cette même fiche et que nous notons avec un tilde pour éviter toute confusion !). Cette fiche est destinée à vous aider à vérifier le résultat de vos programmes en les testant avec $\widetilde{u_0}$ au lieu de u_0. Vous indiquerez vos réponses (correspondant à votre u_0) sur la seconde et vous la remettrez à l'examinateur à la fin de l'épreuve.

En ce qui concerne la partie orale de l'examen, lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures !

Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre n, on demande l'ordre de grandeur en fonction du paramètre, par exemple : $O(n^2)$, $O(n \log n)$,...

Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main ou bien à l'aide de la fiche réponse type fournie en annexe***. Enfin, il est recommandé de lire l'intégralité du sujet avant de commencer afin d'effectuer les bons choix de structures de données dès le début.

1 Préliminaires

Étant donné un corpus, c'est-à-dire une liste de mots valides, on considère un jeu à deux joueurs où, à tour de rôle, chaque joueur énonce une lettre. À eux deux, ils constituent ainsi un mot commun en ajoutant la lettre qu'ils jouent à la fin du mot en cours de construction. Le mot que les deux joueurs construisent doit toujours être un préfixe d'un des mots du corpus. Toutefois, le premier joueur qui forme un mot du corpus en ajoutant sa lettre a perdu. C'est le joueur 1 qui commence. Par exemple, supposons que le corpus est réduit à un seul mot, {ARBORE}. Une partie peut alors se dérouler comme suit : le joueur 1 dit la lettre A, le joueur 2 la lettre R, puis ils jouent alternativement B, O, R, et E. À ce point, le joueur 2 perd à l'instant où il joue la lettre E, puisqu'il a formé le mot ARBORE. Si par contre le corpus contenait également le mot A, alors le joueur 1 perdrait dès son premier coup en prononçant la lettre A.

Notations

On rappelle que pour deux entiers naturels a et b, $a \mod b$ désigne le reste de la division entière de a par b, c'est à dire l'unique entier r avec $0 \leq r < b$ tel que $a = k \times b + r$ pour $k \in \mathbb{N}$.

Rappels sur les arbres lexicographiques et les jeux

Un mot w de taille n sur un alphabet Σ est constitué de n lettres $w[0] \cdots w[n-1]$, toutes dans Σ. ε désigne le mot vide. $p \subseteq w$ indique que p est un préfixe de w. Si de plus $p \neq w$ on note $p \subset w$ et on dit que p est un préfixe strict de w.

Un arbre lexicographique, ou arbre de préfixes, est un arbre de recherche qui permet de représenter un ensemble de mots, et supporte des opérations de recherche et d'insertion. On décrit un tel arbre par le quintuplet $A = (\Sigma, V, \alpha, \delta, T)$ où :

- Σ est un alphabet, ici représenté par les entiers de 0 à $M-1$;
- V représente les nœuds de l'arbre, ici des entiers dont 0 est la racine ;
- $\alpha : V \to \mathcal{P}(\Sigma)$ représente la fonction de voisinage et $\delta : V \times \Sigma^* \to V$ la fonction de transition dans l'arbre : s'il y a une arête étiquetée $a \in \Sigma$ du nœud u au nœud v, alors $a \in \alpha(u)$ et $v = \delta(u, a)$. On étend la fonction de transition δ aux mots de la façon suivante : $\delta(u, \varepsilon) = u$ et $\delta(u, ww') = \delta(\delta(u, w), w')$ où ww' désigne la concaténation des mots w et w'.
- $T \subseteq V$ représente les nœuds dits terminaux de l'arbre.

On peut associer un mot à chaque nœud de l'arbre : la racine de l'arbre correspond au mot vide ε, et on obtient le mot d'un nœud en lisant les lettres sur la branche reliant la racine à ce nœud, cf. Figure 1. L'ensemble des mots de l'arbre correspond aux mots des nœuds terminaux. Et les mots des nœuds de l'arbre correspondent aux préfixes des mots de l'arbre. Pour savoir si un mot w est dans l'arbre, on part de la racine et on emprunte les arêtes étiquetées par les lettres de w si elles existent. S'il ne manque aucune arête et qu'on aboutit à un nœud $v \in T$, alors le mot est dans l'arbre. Dans le cas contraire, le mot n'est pas dans l'arbre.

Un jeu à deux joueurs sur un arbre $A = (\Sigma, V, \alpha, \delta, T)$ se déroule de la façon suivante :

- les nœuds V sont les configurations possibles du jeu ; la racine est contrôlée par le joueur 1, et une arête relie toujours deux nœuds contrôlés par des joueurs différents ;
- les arêtes représentent les coups possibles depuis le nœud u : si l'on joue la lettre $a \in \alpha(u)$ depuis le nœud u, on emprunte l'arête et on atterrit dans le nœud $v = \delta(u, a)$;

— le jeu s'arrête lorsqu'on atteint un nœud de T, dans ce cas l'un des joueurs gagne.

Générateur de nombres pseudo-aléatoires

Étant donné u_0 on définit la récurrence suivante :

$$u(0) = u_0$$
$$\forall t \in \mathbb{N},\ u(t+1) = (28 \times u(t)) \mod 336\,529$$

L'entier u_0 vous est donné, et doit être recopié sur votre fiche réponse avec vos résultats. Une fiche réponse type vous est donnée en exemple, et contient tous les résultats attendus pour une valeur de u_0 différente de la vôtre (notée $\widetilde{u_0}$). Il vous est conseillé de tester vos algorithmes avec cet $\widetilde{u_0}$. Pour chaque calcul demandé, avec le bon choix d'algorithme le calcul ne devrait demander qu'au plus quelques secondes, jamais plus d'une minute.

Question 1 *Calculer les valeurs suivantes :*

a) $u(1) \mod 1\,000$, ***b)*** $u(42) \mod 1\,000$, ***c)*** $u(10^5) \mod 1\,000$.

Génération de mots

On définit le corpus $L(N)$ par la liste des mots w_i pour $i = 0, \ldots, N-1$ comme suit : w_i est de taille T_i et sa t-ième lettre est $w_i[t]$ pour $t = 0, \ldots, T_i - 1$ où :

$$T_i = T_{\min} + (u(i) \mod (T_{\max} - T_{\min} + 1)) \quad w_i[t] = u(\lfloor i/7 \rfloor + t) \mod M$$

Les mots ont au moins $T_{\min} = 4$ lettres, au plus $T_{\max} = 24$ lettres et l'alphabet comporte $M = 17$ lettres de 0 à 16. En particulier, $\varepsilon \notin L(N)$.

Question 2 *Afficher les 4 dernières lettres du dernier mot w_{N-1} du corpus $L(N)$ pour les valeurs de N suivantes :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

On note $\textsc{NbOcc}(a, L)$ le nombre d'occurrences d'une lettre $a \in \Sigma$ dans le corpus L, dont on se servira dans la dernière section 3. Il se peut qu'un même mot apparaisse plusieurs fois dans le corpus, auquel cas les lettres d'un mot apparaissant plusieurs fois doivent être comptées plusieurs fois.

Question 3 *Calculer $\textsc{NbOcc}(w_{N-1}[T_{N-1}-1], L(N))$, le nombre d'occurrences de la dernière lettre du dernier mot du corpus $L(N)$, pour les valeurs de N suivantes :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

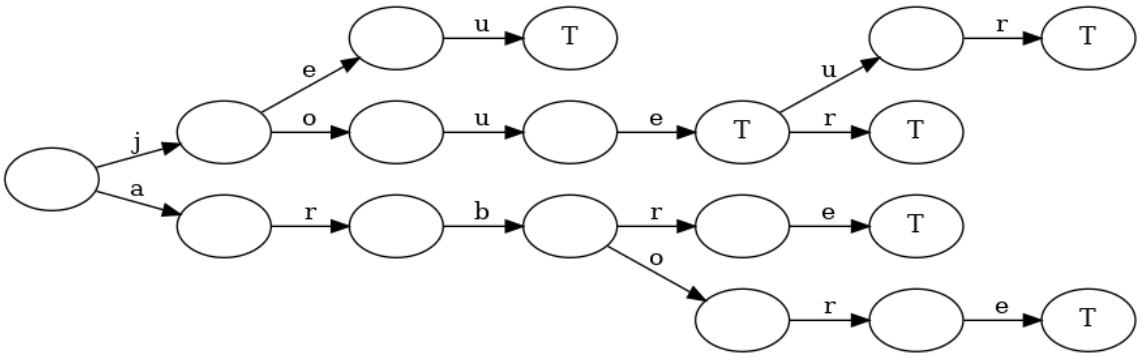

FIGURE 1 – Un arbre lexicographique pour le corpus {arbore, arbre, jeu, joue, jouer, joueur}. La racine est le nœud tout à gauche et les nœuds terminaux sont indiqués par T.

Construction de l'arbre lexicographique

On construit un arbre lexicographique associé à un corpus en partant d'un arbre vide et en insérant successivement les mots du corpus dans l'arbre, comme représenté sur la Figure 1. On note $A(N)$ l'arbre lexicographique associé au corpus $L(N)$.

Question 4 *Pour les valeurs de N suivantes, déterminer le nombre de nœuds terminaux présents dans $A(N)$.*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 1 *Décrire la structure de donnée utilisée pour représenter un arbre lexicographique et la complexité de l'algorithme utilisé pour le construire, en fonction de N, $T_{\max}$ et M. Quel est l'intérêt d'une telle structure de données, et quelle autre structure pourrait-on utiliser pour stocker le corpus ? À quoi correspond le nombre de nœuds terminaux ?*

2 Stratégie gagnante contre un adversaire omniscient

Pour commencer, on suppose dans cette section que vous et votre adversaire connaissez tous les mots du corpus, que c'est vous qui commencez (vous êtes le joueur 1) et que l'adversaire a la meilleure stratégie possible.
On note MOTSINACCESSIBLES(N) la liste des mots distincts du corpus $L(N)$ qu'on ne pourra jamais jouer, car le jeu s'interrompt forcément avant d'avoir pu les prononcer.

Question 5 *Compter la taille de* MOTSINACCESSIBLES(N) *pour les valeurs de N suivantes :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 2 *Présenter votre algorithme et sa complexité en fonction de N, $T_{\max}$ et M.*

Dans l'arbre, un nœud est dit gagnant si et seulement si le joueur 1 (vous, dans cette section) a une stratégie pour gagner à partir de ce nœud, quoi que fasse le joueur 2. Notez que "gagnant" se réfère toujours au joueur 1 qui commence, même si c'est au joueur 2 de jouer à partir de ce nœud. On dit que le joueur 1 (vous) a une stratégie gagnante si la racine est gagnante. On vous garantit que la racine est gagnante pour le corpus généré.

Question à développer pendant l'oral 3 *Formellement, dans quels cas un nœud est-il gagnant ?*

Question 6 *Calculer le nombre de nœuds gagnants qui sont des voisins directs de la racine dans le corpus $L(N)$ pour les valeurs de N suivantes :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 4 *Présenter votre algorithme et sa complexité en fonction de N, $T_{\max}$ et M.*

À présent que vous avez vérifié que le joueur 1 a une stratégie gagnante, on s'intéresse à la stratégie la plus concise, i.e. retenir un nombre minimum de mots pour pouvoir gagner au jeu.
Un ensemble de mots est stable si, en ne jouant que des lettres correspondant à des préfixes de ces mots, on s'assure que l'adversaire ne peut que jouer des préfixes de ces mots. D'une certaine manière, un tel ensemble piège l'adversaire. Formellement, un ensemble L' est stable si et seulement si pour tout mot w de L' et tout préfixe strict $p \subset w$ de taille impaire, si $v = \delta(0, p)$, alors on a la relation suivante : $\forall b \in \alpha(v), \exists w' \in L', pb \subseteq w'$.
Par exemple, pour le corpus $[\text{ET}, \text{EN}, \text{BAS}]$, $\{\text{ET}, \text{EN}\}$ et $\{\text{BAS}\}$ sont stables mais $\{\text{ET}\}$ n'est pas stable.
On cherche alors un ensemble stable de taille minimale qui ne contient que des mots qui nous font gagner : un ensemble de mots distincts Antisèche(N) du corpus $L(N)$ de cardinal minimal tel que jouer seulement avec les mots de Antisèche(N) nous permette de gagner.

Question 7 *Calculer la taille de* Antisèche(N) *pour les valeurs de N suivantes :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 5 *Présenter votre algorithme et sa complexité en fonction de N, $T_{\max}$ et M.*

3 Pas de stratégie gagnante contre un adversaire limité

Dans cette section, c'est l'adversaire qui commence : c'est lui le joueur 1. Heureusement, il n'est pas omniscient et même assez limité, ainsi vous vous intéressez au meilleur coup à jouer à partir d'un nœud donné. Ainsi dans cette section, on ne s'intéresse plus à gagner à coup sûr, mais à quantifier la valeur d'un nœud, d'un coup (i.e. une paire nœud-arête), d'une stratégie.
On introduit un nœud supplémentaire $\perp$ qui n'a pas de coup possible et on note $V^{\perp} = V \cup \{\perp\}$.

Déroulement d'une partie

On appelle politique une manière de jouer. Formellement, une politique $\pi(a|u)$ est la probabilité que l'on a de jouer la lettre $a \in \Sigma$ en étant dans le nœud u. Dans cette section nous allons étudier comment créer une séquence de politiques $\pi_0, \ldots, \pi_\ell$ où π_{i+1} est une amélioration de π_i. Au départ, vous ne savez pas quel coup jouer et jouez de façon uniformément aléatoire $\pi_0 = \pi_{\text{unif}}$ parmi les coups possibles depuis chaque nœud. Au fur et à mesure de l'apprentissage vous améliorerez votre politique.
Les sessions $t = 1, \ldots, T$ correspondent à votre tour de jeu. Une session contient deux tours, joueur 2 puis joueur 1, et se déroule de la façon suivante : à partir d'un nœud u, on peut jouer une lettre $a \in \alpha(u)$ qui aboutit au nœud $v = \delta(u, a)$ de l'adversaire. Si l'on perd immédiatement on atterrit dans $\perp$. Sinon, l'adversaire peut jouer une lettre supplémentaire b, nous fait atterrir dans le nœud $u' = \delta(v, b) = \delta(u, ab)$, et on passe à la session suivante. Dans toute cette section, lorsque l'adversaire doit jouer, il emprunte l'arête proportionnellement au nombre d'occurrences de la lettre dans le corpus :

$$p(b|v) = \frac{\textsc{NbOcc}(b, L)}{\sum_{b' \in \alpha(v)} \textsc{NbOcc}(b', L)}$$

où $p(b|v)$ est la probabilité que l'adversaire joue la lettre b en partant du nœud v.

Question 8 *Pour les valeurs de N suivantes, calculer la lettre ℓ_0 du premier coup le plus probable joué par l'adversaire ; en cas d'égalité, donner la plus petite lettre.*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Mécanisme de récompense

Au début de chaque session, le jeu est sur un nœud u où l'on doit jouer, et il faut choisir une lettre $a \in \alpha(u)$ qui amène le jeu dans le nœud $v = \delta(u,a)$. Si l'on forme un mot du corpus, alors, pour cette section seulement, on dit que l'adversaire amène le jeu dans l'état spécial $u' = \perp$ avec une récompense $r = -1$. Sinon la partie continue et l'adversaire choisit une lettre $b \in \alpha(v)$ selon la loi de probabilité $p(b|v)$ et amène le jeu dans l'état $u' = \delta(v,b)$ avec une récompense qui vaut $r = 1$ s'il forme un mot du corpus et $r = 0$ sinon, auquel cas la partie continue et on passe à la session suivante.
On note $p(u',r|v)$ la probabilité d'atterrir dans le nœud u' avec une récompense r en partant de $v = \delta(u,a)$ où a est la lettre qu'on joue depuis le nœud u.

Question à développer pendant l'oral 6 *Pour quelles valeurs de $(u',r,v) \in V^\perp \times \{-1, 0, 1\} \times V$ est-ce que $p(u',r|v)$ est non nul ?*

Détaillons à présent ce que l'on cherche à optimiser :
- U_t est une variable aléatoire correspondant au nœud à partir duquel on joue à l'instant t, A_t la lettre jouée à l'instant t qui aboutit au nœud U_{t+1} (à partir duquel on joue à l'instant $t+1$) avec une récompense de R_{t+1} ;
- γ est un facteur dit d'actualisation pour les récompenses futures, dans tout ce sujet $\gamma = 0{,}9$;

— G_t est la récompense amortie : $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$. Ainsi cette valeur objectif accorde plus d'importance aux récompenses proches qu'aux récompenses lointaines dans le temps. Intuitivement, on souhaite maximiser nos chances de gagner dans un futur proche.

De plus pour une politique π on note :

— la fonction de valeur $\mathcal{V}_\pi$ sur les nœuds, qui représente la récompense amortie moyenne si l'on part du nœud u et que l'on suit la politique π : $\mathcal{V}_\pi(u) = \mathbb{E}_\pi[G_t | U_t = u]$;

— la fonction de valeur $\mathcal{Q}_\pi$ sur les coups (nœuds-arêtes), qui représente la récompense amortie moyenne si l'on part du nœud u, que l'on joue la lettre a et que l'on suit la politique π : $\mathcal{Q}_\pi(u, a) = \mathbb{E}_\pi[G_t | U_t = u, A_t = a]$.

Évaluation de politique

On définit l'opérateur de Bellman $\mathcal{K}_\pi$:

$$\mathcal{K}_\pi(V)(u) = \sum_{a \in \alpha(u)} \left(\pi(a|u) \sum_{(u',r) \in V^\perp \times \{-1,0,1\}} p(u',r|\delta(u,a))[r + \gamma V(u')] \right).$$

L'opérateur $\mathcal{K}_\pi$ est contractant et admet un point fixe qui est $\mathcal{V}_\pi$ (on ne vous demande pas de le prouver). Ainsi on définit la suite $V_\pi^{(0)}(u) = 0$ pour tout u et $V_\pi^{(i+1)} = \mathcal{K}_\pi(V_\pi^{(i)})$, cette suite tend vers $\mathcal{V}_\pi$ et on supposera que $V_\pi^{(100)} = \mathcal{V}_\pi$.

Question 9 *Pour les valeurs de N suivantes, calculer $V_{\pi_{\text{unif}}}^{(100)}(u'_0)$ où π_{unif} est la politique initiale uniformément aléatoire et u'_0 est le nœud à partir duquel on joue si l'adversaire commence en jouant ℓ_0 :*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 7 *Donner une borne supérieure sur le nombre d'itérations nécessaires à la convergence. En déduire un autre algorithme pour calculer $\mathcal{V}_\pi$ et sa complexité.*

Amélioration de politique

À présent qu'on a calculé la valeur de chaque nœud, on peut améliorer notre politique :

$$\pi_{i+1}(a|u) = \operatorname*{argmax}_a \mathcal{Q}_{\pi_i}(u, a) \triangleq \operatorname*{argmax}_a \sum_{(u',r) \in V^\perp \times \{-1,0,1\}} p(u',r|\delta(u,a))[r + \gamma \mathcal{V}_{\pi_i}(u')]$$

À nouveau, en cas d'égalité, on préférera la plus petite lettre. Ce qui signifie que $\pi_{i+1}(a|u)$ vaut 1 ssi a est la plus petite lettre vérifiant $\mathcal{Q}_{\pi_i}(u, a) = \max_{a'} \mathcal{Q}_{\pi_i}(u, a')$, et 0 partout ailleurs.

Question 10 *Alterner entre évaluation de $\mathcal{V}_\pi$ puis amélioration de π jusqu'à convergence. Si l'adversaire joue ℓ_0 et qu'on atterrit en u'_0, quel est le meilleur coup à jouer pour les valeurs suivantes de N ?*

a) $N = 10$ ***b)*** $N = 1\,000$, ***c)*** $N = 100\,000$.

Question à développer pendant l'oral 8 *Expliquer l'algorithme utilisé et donner sa complexité par itération en fonction de N, $T_{\max}$, M.*

Question à développer pendant l'oral 9 *Si le nombre de nœuds est fini et petit, et qu'on considère un graphe quelconque de jeu, quel algorithme proposez-vous pour déterminer la fonction de valeur sur les nœuds $\mathcal{V}_\pi$?*

Fiche réponse type : Jeux sur des arbres (*tries on tries*).

$\widetilde{u_0}$: 42

Question 1

a) 176

b) 338

c) 645

Question 2

a) (3, 16, 12, 5)

b) (1, 11, 15, 12)

c) (13, 6, 14, 13)

Question 3

a) 10

b) 1395

c) 79680

Question 4

a) 8

b) 883

c) 86502

Question 5

a) 6

b) 740

c) 72795

Question 6

a) 2

b) 3

c) 16

Question 7

a) 1

b) 1

c) 17

Question 8

a) 3

b) 0

c) 5

Question 9

a) -0.9

b) 0.10435984226959871

c) -0.12291525213759016

Question 10

a) 16

b) 5

c) 8

Fiche réponse : Jeux sur des arbres (*tries on tries*).

Nom, prénom, u_0 : .....

Question 1

a)

b)

c)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

a)

b)

c)

Question 6

a)

b)

c)

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

c)

Question 10

a)

b)

c)