Election de leader

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

Durée de l'épreuve: 3 heures 30 minutes

Juin/Juillet 2018

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.

Soient $m = 2^{31} - 1 = 2\,147\,483\,647$, $x = 16\,807$ et $y = 17$. On définit la suite $u_i = (xu_{i-1} + y) \bmod m$ pour tout $i \geq 1$, où u_0 est donné sur votre fiche. Ici $a \bmod b$ désigne le reste de la division euclidienne de a par b.

Question 1 *Calculer* ***a)*** $u_1 \bmod 1000$ ***b)*** $u_2 \bmod 1000$ ***c)*** $u_{50} \bmod 1000$ ***d)*** $u_{999\,999} \bmod 1000$.

Dans ce sujet, on s'intéresse aux algorithmes distribués qui sont exécutés par un ensemble de machines qui communiquent entre elles par envoi de messages ou par variable partagée. La possibilité de communication entre deux machines est spécifiée par un graphe : les sommets représentent les machines, et les arêtes la possibilité de communication entre une paire de machines. Nous allons étudier deux algorithmes qui permettent de distinguer une unique machine qui devient le leader dans le réseau.

Les deux sections suivantes sont indépendantes.

1 Election dans un graphe

Étant donnés $n, p, a, b \geq 1$, on définit le graphe non-orienté $G(n, p, a, b) = (S, A)$ sur les sommets $S = \{0, 1, \ldots, n-1\}$ et les arêtes A comme suit. Pour tout $0 \leq i < j \leq n-1$, il existe une arête entre i et j, c'est-à-dire $\{i, j\} \in A$, si et seulement si

$$(u_{g(a,i,b,j)} \bmod p) \leq 1,$$

où $g(a, i, b, j) = (a(i+1) + b(j+1)) \bmod 10^6$.
Ainsi, dans ce graphe, un sommet j est voisin d'un sommet i si et seulement si $j \neq i$ et
- soit $i < j$ et $(u_{g(a,i,b,j)} \bmod p) \leq 1$,
- soit $j < i$ et $(u_{g(a,j,b,i)} \bmod p) \leq 1$.

Question 2 *Donner le nombre de voisins du sommet d'identifiant $n/2$ dans les graphes $G(n, p, a, b)$ suivants.* ***a)*** $n = 10, p = 5, a = 1, b = 1$ ***b)*** $n = 100, p = 10, a = 2, b = 2$ ***c)*** $n = 500, p = 20, a = 3, b = 3$

Question à développer pendant l'oral 1 *Décrire l'algorithme de parcours en largeur sur les graphes et analyser sa complexité.*

On considère un réseau de machines associé au graphe $G = (S, A)$ défini comme suit : à chaque sommet i se trouve une machine d'identifiant i qui peut communiquer avec une machine j si et seulement si $\{i, j\} \in A$. On dit que j est un voisin de i.
Chaque machine dispose d'une file qui stocke les messages reçus par ses voisins dans l'ordre d'arrivée.
Nous allons étudier un algorithme distribué qui permet de simuler un parcours en largeur dans le graphe associé au réseau de machines. Dans cet algorithme, le parcours est lancé par une machine d'indice i_0 donné. Chaque machine dispose d'une variable `distance` qui vaut initialement ∞ pour toutes les machines $i \neq i_0$ et qui vaut 0 pour la machine i_0. Par ailleurs, chaque machine a une variable entière `parent` qui est initialement non-définie.
L'algorithme fonctionne par étapes. À l'étape 0, la machine i_0 envoie à tous ses voisins la valeur de sa variable `distance` (qui est ajoutée dans la file de réception de ses voisins). À l'étape $t \geq 1$, la machine $(t \bmod n)$ exécute le code suivant.

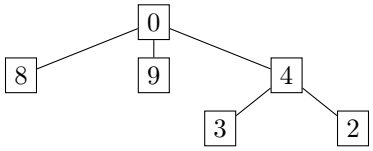
Si la file de messages est non-vide :
- Enlever un seul message d de la tête de la file
- Si $d+1 <$ `distance` alors
  - `distance` $\leftarrow d+1$
  - Envoyer `distance` à tous les voisins
  - `parent` $\leftarrow j$, où j est la machine originaire du message d

On dit que le temps de terminaison est t si l'étape t est la dernière étape où une machine a une file de message non vide.

Question 3 *Calculer le temps de terminaison de l'algorithme sur les graphes $G(n,p,a,b)$ suivants pour $i_0 = 0$. On donnera le résultat modulo 1000.* ***a)*** $n = 10, p = 5, a = 1, b = 1$ ***b)*** $n = 50, p = 10, a = 2, b = 2$ ***c)*** $n = 1000, p = 50, a = 3, b = 3$ ***d)*** $n = 5000, p = 100, a = 4, b = 4$

Question à développer pendant l'oral 2 *Démontrer que les variables* `distance` *appartiennent à $\{\infty\} \cup \{0, 1, \ldots, n\}$. En déduire un majorant sur le nombre de messages envoyés dans l'algorithme et analyser la complexité.*

On considère les arbres généraux qui sont des arbres enracinés dans lesquels les sommets peuvent avoir un nombre quelconque de fils. Le graphe suivant est un exemple d'arbre général dont la racine est le sommet 0.



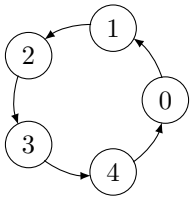
On suppose que l'algorithme précédent a été exécuté sur un graphe $G = (S, A)$. Pour chaque sommet s, on pose $\mathsf{fils}(s) = \{t \in S \mid t.\texttt{parent} = s\}$ où $t.\texttt{parent}$ est la variable `parent` de la machine t. La structure ainsi définie est un arbre dont la racine est le sommet i_0 si c'est la machine i_0 qui a lancé le parcours. Pour un graphe G, on note $T(G)$ l'arbre ainsi obtenu après l'exécution de l'algorithme avec $i_0 = 0$.
On définit la valeur d'un sommet s de l'arbre $T(G)$, notée $\mathsf{val}(s)$, récursivement comme suit.
- Si s est une feuille, alors $\mathsf{val}(s) = s$,
- Sinon, $\mathsf{val}(s) = s + \min_{t \in \mathsf{fils}(s)} \mathsf{val}(t)$.

Par exemple, la valeur du sommet 0 dans l'exemple précédent est 6.

Question 4 *Calculer* $\mathsf{val}(0)$ *dans les arbres $T(G(n,p,a,b))$ pour les paramètres suivants.* ***a)*** $n = 10, p = 5, a = 1, b = 1$ ***b)*** $n = 50, p = 10, a = 2, b = 2$ ***c)*** $n = 1000, p = 50, a = 3, b = 3$ ***d)*** $n = 5000, p = 100, a = 4, b = 4$

2 Election dans un anneau

On considère désormais les réseaux de machines définis par un graphe sous forme d'anneau orienté comme l'exemple suivant :



Dans l'algorithme que nous allons étudier, chaque machine dispose d'une variable contenant un bit. Chaque machine peut lire et écrire sur sa propre variable, et lire la variable de la machine précédente.

Formellement, on identifie les machines par leur indice $0, \dots, n-1$, et on note b_i le bit de la machine i. On considère les indices des machines modulo n, par exemple $b_{-1} = b_{n-1}$ ou encore, $b_n = b_0$. Ainsi, la machine i peut lire b_{i-1} et b_i et écrire b_i. On définit $\mathcal{C}_n = \{0,1\}^n$ l'ensemble de configurations du réseau à n machines, où une configuration consiste à spécifier la valeur de la variable de chaque machine. On note une configuration par un vecteur $\vec{b} \in \mathcal{C}_n$, où b_i désigne le i-ème élément.

On dit qu'une machine i a un jeton si $b_{i-1} = b_i$. Par exemple, dans la configuration $(0,1,0,0)$, les machines 0 et 3 ont un jeton.

L'algorithme fonctionne par étapes. À chaque étape, si on est à la configuration $\vec{b} \in \mathcal{C}_n$, une machine i est sélectionnée par un ordonnanceur et on passe à la configuration $\vec{b'} = \mathsf{Succ}(\vec{b}, i) \in \mathcal{C}_n$ définie comme suit :

— pour tout $j \neq i$, $b'_j = b_j$,
— on a $b'_i = 1 - b_i$ si $b_i = b_{i-1}$ et $b'_i = b_i$ sinon.

Ainsi, une seule machine exécute une action à chaque étape, en inversant son bit si et seulement si celui-là est égal au bit de la machine précédente.

Par exemple, à partir de la configuration $(0,1,0,0)$ on passe à la configuration $(1,1,0,0)$ si c'est la machine 0 qui est sélectionnée. Si c'est la machine 3 qui est sélectionnée, alors on passe à $(0,1,0,1)$. Enfin, si c'est la machine 1 qui est sélectionnée, on reste à la configuration $(0,1,0,0)$.

Un ordonnanceur est une fonction $f : \{0,1\}^n \to \{0,1,\dots,n-1\}$ qui à chaque configuration du réseau associe l'indice de la machine sélectionnée. Soit l'ordonnanceur minim qui, étant donnée une configuration $\vec{b}$, sélectionne l'indice $0 \leq i \leq n-1$ minimal tel que $b_{i-1} = b_i$ si un tel indice existe, et 0 sinon.

Étant donnés $n, p \geq 1$, on définit la configuration $\mathcal{B}(n,p,\alpha,\beta) = b_0 b_1 \dots b_{n-1}$ comme suit. On pose $b_0 = 0$, et pour tout $1 \leq i \leq n-1$, on a

$$b_i = \begin{cases} b_{i-1} & \text{si } (u_{h(\alpha,i,\beta)} \bmod p) \leq 1, \\ 1 - b_{i-1} & \text{sinon.} \end{cases}$$

où $h(\alpha,i,\beta) = (\alpha i + \beta) \bmod 10^6$.

Question 5 *Pour les paramètres suivants, donner le plus petit indice $\beta \geq 1$ tel que* minim *renvoie un indice supérieur ou égal à 3 pour la configuration $\mathcal{B}(n,p,\alpha,\beta)$:* ***a)*** $n = 10, p = 10, \alpha = 1$ ***b)*** $n = 100, p = 10, \alpha = 2$ ***c)*** $n = 1000, p = 30, \alpha = 3$ ***d)*** $n = 10\,000, p = 100, \alpha = 10$

Une exécution à partir de $\vec{b}^1 \in \mathcal{C}_n$ avec un ordonnanceur f est une suite $(\vec{b}^t)_{t\geq 1}$ où $\vec{b}^{t+1} = \mathsf{Succ}(\vec{b}^t, f(\vec{b}^t))$. Il s'agit de la suite de configurations obtenue étapes par étapes en exécutant la machine sélectionnée par f.
La suite suivante est le début de l'exécution à partir de $\vec{b}^1 = (0,1,0,0)$ avec minim :

$$(0,1,0,0) \to (1,1,0,0) \to (1,0,0,0) \to (1,0,1,0) \to (1,0,1,0) \to \dots$$

On dit qu'une configuration est stable si elle contient au plus un jeton. Si la configuration stable ne contient pas de jeton, on dit qu'aucune machine n'a été élue, sinon la machine qui détient le jeton devient le leader.
Le temps de stabilisation d'une configuration initiale $\vec{b}^1$ pour un ordonnanceur f, est l'indice minimal t tel que $\vec{b}^t$ est une configuration stable où $(\vec{b}^t)_{t\geq 1}$ est l'exécution à partir de $\vec{b}^1$ avec f.

Question 6 *Calculer le temps de stabilisation de la configuration $\mathcal{B}(n,p,\alpha,1)$ avec l'ordonnanceur* minim *pour les paramètres suivants. On donnera le résultat modulo 1000.* ***a)*** $n = 11, p = 5, \alpha = 1$ ***b)*** $n = 100, p = 10, \alpha = 2$ ***c)*** $n = 1001, p = 10, \alpha = 3$ ***d)*** $n = 20\,000, p = 200, \alpha = 4$ ***e)*** $n = 100\,000, p = 100, \alpha = 5$

Question à développer pendant l'oral 3 *Décrire l'algorithme que vous avez utilisé pour la question précédente, et analyser la complexité de l'opération qui effectue une étape d'exécution.*

À chaque configuration $\vec{b}$, on associe son codage noté $\mathsf{code}(\vec{b})$ comme suit : si $0 \leq i_1 < i_2 < \dots < i_k \leq n-1$ sont les indices des machines qui contiennent un jeton, alors on pose $\mathsf{code}(\vec{b}) = (i_2 - i_1, i_3 - i_2, \dots, i_k - i_{k-1})$. On ordonne les codages par ordre lexicographique défini comme suit. Pour deux suites $(a_1, a_2, \dots, a_p)$ et $(b_1, b_2, \dots, b_q)$, on a $(a_1, a_2, \dots, a_p) \prec_{\text{lex}} (b_1, b_2, \dots, b_q)$ si et seulement si soit $p < q$, soit $p = q$ et il existe i_0 tel que pour tout $1 \leq i < i_0$, $a_i = b_i$ et $a_{i_0} < b_{i_0}$.

Question à développer pendant l'oral 4 *Soient deux configurations $\vec{b}, \vec{b'} \in \mathcal{C}_n$ telles que $\vec{b'} = \mathsf{Succ}(\vec{b}, \mathsf{minim}(\vec{b}))$. Démontrer que $\mathsf{code}(\vec{b'}) \prec_{lex} \mathsf{code}(\vec{b})$. Conclure.*

Question à développer pendant l'oral 5 *Existe-t-il des configurations qui n'atteignent jamais une configuration stable avec certains ordonnanceurs ?*

Etant donnée une configuration de départ, on se propose maintenant de calculer l'ordonnanceur le plus rapide, c'est-à-dire, celui qui atteint une configuration stable en un nombre d'étapes minimal. Formellement, le temps de stabilisation minimal d'une configuration est l'entier naturel minimal t tel qu'il existe un ordonnanceur dont le temps de stabilisation pour cette configuration est égal à t.

Question 7 *Donner le temps de stabilisation minimal des configurations $\mathcal{B}(n,p,\alpha,\beta)$ suivantes.* ***a)*** $n = 10, p = 5, \alpha = 1, \beta = 1$ ***b)*** $n = 20, p = 15, \alpha = 2, \beta = 2$ ***c)*** $n = 30, p = 25, \alpha = 3, \beta = 3$

Question à développer pendant l'oral 6 *Décrire votre algorithme. Sa complexité est-elle polynomiale ? Justifier. Donner un ordre de grandeur sur n, le nombre de jetons et leurs positions respectives pour lesquels votre algorithme termine rapidement.*

Nous revenons à la question du temps de stabilisation avec l'ordonnanceur minim mais pour les grandes instances :

Question 8 *Donner le temps de stabilisation de la configuration $\mathcal{B}(n, p, \alpha, \alpha)$ avec l'ordonnanceur* minim *pour les paramètres suivants. On donnera le résultat modulo 1000.* ***a)*** $n = 10^6, p = 1001, \alpha = 1$ ***b)*** $n = 10^7, p = 1001, \alpha = 2$ ***c)*** $n = 2 \cdot 10^7, p = 101, \alpha = 3$ ***d)*** $n = 10^8, p = 101, \alpha = 4$

Question à développer pendant l'oral 7 *Décrire votre algorithme et analyser sa complexité en temps et en espace.*

Pour une suite finie d'entiers naturels $A = A_1 A_2 \dots A_k$, soit $\mathsf{cons}(A)$ l'ensemble des pairs consécutives de la suite A, c'est-à-dire :

$$\mathsf{cons}(A) = \{(A_1, A_2), (A_2, A_3), \dots, (A_{k-1}, A_k), (A_k, A_1)\}.$$

Pour un ensemble B d'entiers naturels, on note par $A \setminus B$ la suite obtenue à partir de A en supprimant tout élément qui appartient à B. Par exemple, $\mathsf{cons}(1, 4, 5, 10, 40) = \{(1, 4), (4, 5), (5, 10), (10, 40), (40, 1)\}$ et $(1, 4, 5, 10, 40) \setminus \{4, 10\} = (1, 5, 40)$.
On définit $d_n(i, j) = j - i$ si $i < j$ et $n - i + j$ sinon.
Étant donnée une instance de notre problème, soit A la suite croissante des indices de machines qui contiennent un jeton. On observera que le temps de stabilisation minimal dépend uniquement de n et de A. Ainsi on le note $\mathsf{Opt}_n(A)$.

Question à développer pendant l'oral 8 *On admet que*

$$\mathsf{Opt}_n(A) = \min_{(a,b) \in \mathsf{cons}(A)} d_n(a, b) + \mathsf{Opt}_n(A \setminus \{a, b\}).$$

En déduire un algorithme efficace pour calculer le temps de stabilisation minimale d'une configuration.

Question 9 *Donner le temps de stabilisation minimale de la configuration $\mathcal{B}(n, p, \alpha, \alpha)$. On donnera le résultat modulo 1000.* ***a)*** $n = 10^5, p = 1\,001, \alpha = 1$ ***b)*** $n = 10^5, p = 5\,001, \alpha = 2$ ***c)*** $n = 10^6, p = 10\,001, \alpha = 3$ ***d)*** $n = 10^6, p = 20\,001, \alpha = 4$

Fiche réponse type: Election de leader

$\widetilde{u_0}$: 1

Question 1

a) 824

b) 985

c) 478

d) 930

Question 2

a) 3

b) 26

c) 49

Question 3

a) 56

b) 588

c) 57

d) 136

Question 4

a) 7

b) 6

c) 74

d) 55

Question 5

a) 5

b) 9

c) 2

d) 6

Question 6

a) 3

b) 46

c) 414

d) 144

e) 196

Question 7

a) 5

b) 6

c) 6

Question 8

a) 534

b) 340

c) 500

d) 200

Question 9

a) 272

b) 452

c) 385

d) 420

Fiche réponse: Election de leader

Nom, prénom, u_0: .....

Question 1

a)

b)

c)

d)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

d)

Question 4

a)

b)

c)

d)

Question 5

a)

b)

c)

d)

Question 6

a)

b)

c)

d)

e)

Question 7

a)

b)

c)

Question 8

a)

b)

c)

d)

Question 9

a)

b)

c)

d)