Jeux à plusieurs et coalitions

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

Durée de l'épreuve: 3 heures 30 minutes

Juillet 2005

ATTENTION !

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

Important.
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***.

1 Préambule

Soit $(u_n)_{n\in\mathbb{N}}$ la suite définie par : u_0 est le numéro inscrit sur votre table d'examen, et

$$u_{n+1} = (101u_n + 269) \bmod 4\,890\,367$$

Question 1 *Que valent* ***a)*** u_1 *?* ***b)*** u_{100} *?* ***c)*** $u_{1\,000}$ *?*
d) *Quel est le nombre de* i *avec* $1 \leq i \leq 4\,999$ *tels que*

$$u_i + u_{5\,000-i} \quad \leq \quad u_1 + u_{4\,999} \text{ ?}$$

2 Opérations ensemblistes

Définir une structure de données permettant de représenter des ensembles finis de n-uplets d'entiers naturels. (L'entier n variera d'un ensemble à l'autre, mais deux éléments d'un même ensemble seront toujours des n-uplets avec le même n inférieur ou égal à 6.) On pourra par exemple représenter l'ensemble fini $\{\vec{x}_1, \ldots, \vec{x}_k\}$ sous forme d'une liste, triée ou non, contenant $\vec{x}_1$, …, $\vec{x}_k$ sans répétition ($\vec{x}_i \neq \vec{x}_j$ pour tous $i \neq j$). En général, le choix de la structure de données utilisée est laissée à l'appréciation du candidat.

Question 2 *En utilisant cette structure de données, écrire des fonctions :*
- `s_egal` *prenant deux ensembles finis en argument, et retournant vrai si et seulement si les deux ensembles sont égaux. Par exemple, si les deux ensembles sont codés sous forme de deux listes* $[\vec{x}_1, \ldots, \vec{x}_k]$ *et* $[\vec{x}'_1, \ldots, \vec{x}'_{k'}]$*, ils sont égaux si et seulement si la deuxième liste est une permutation de la première.*
- `s_vide` *calculant l'ensemble vide.*
- `s_singl` *prenant un élément* $\vec{x}$ *et retournant le singleton* $\{\vec{x}\}$*.*
- `s_union` *calculant l'union de deux ensembles en argument.*
- `s_inter` *calculant l'intersection de deux ensembles en argument.*
- `s_card` *calculant le cardinal de l'ensemble en argument.*

(Vous pourrez avoir à en écrire d'autres dans la suite.)

★ *Vous présenterez à l'oral les algorithmes que vous avez utilisés, ainsi que leurs complexités.*

Question 3 *Testez vos programmes en répondant aux questions suivantes. Combien y a-t-il d'éléments dans*
a) $\{(0,0,1)\} \cup \{(0,2,1)\}$ *?*
b) $\{(0,0,1)\} \cup \{(0,0,1)\}$ *?*
c) *l'intersection des deux ensembles précédents ?*

Question 4 *Combien y a-t-il d'éléments dans*
a) A*, défini par* $A = \{(x,y,z) \mid 0 \leq x,y,z \leq 10, x^2 + y^2 = z^2\}$ *?*
b) B*, défini par* $B = \{(u_{3n} \bmod 11, u_{3n+1} \bmod 11, u_{3n+2} \bmod 11) \mid 0 \leq n \leq 100\}$ *?*
c) *l'intersection* $A \cap B$ *des deux ensembles précédents ?*
d) *Lister explicitement les éléments de* $A \cap B$*.*

3 Jeux

On considère des jeux, définis comme suit. Un jeu est un quadruplet $(\mathcal{J}, \mathcal{C}, t, \delta)$, où :
- $\mathcal{J}$ est un ensemble fini, dont les éléments sont appelés les joueurs.
- $\mathcal{C}$ est un ensemble dit de positions du jeu.
- La fonction $t : \mathcal{C} \to \mathcal{J}$ attribue à chaque position le joueur dont c'est le tour de jouer.
- La fonction $\delta : \mathcal{C} \to \mathbb{P}(\mathcal{C})$ associe à chaque position c l'ensemble $\delta(c)$, possiblement vide, des positions vers lesquelles le joueur $t(c)$ peut aller en jouant un coup du jeu.

Un exemple est le jeu de Nim. Dans ce jeu, deux joueurs doivent enlever à tour de rôle une à trois allumettes d'un tas d'allumettes donné commun aux deux joueurs. Le premier joueur qui n'a plus d'allumette à enlever perd. Ici, l'ensemble des joueurs $\mathcal{J}$ est l'ensemble $\{1, 2\}$, et l'ensemble des positions $\mathcal{C}$ est l'ensemble des couples (j, k), où $j \in \mathcal{J}$ est le numéro du joueur dont c'est le tour de jouer, et $k \in \mathbb{N}$ est le nombre d'allumettes restantes. La fonction t envoie (j, k) vers j, et on a :

$$\delta(j, k) = \{(j + 1 \bmod 2, k') | k' \in \mathbb{N}, k - 3 \leq k' \leq k - 1\}.$$

En général, un jeu peut avoir plus de deux joueurs. Le jeu que nous allons étudier est une généralisation du jeu de Nim, noté $Nim(N, n, K, r)$, où $N, n, K, r \in \mathbb{N}$ sont des paramètres. Il contient N joueurs notés $1, 2, \ldots, N$; autrement dit $\mathcal{J} = \{1, 2, \ldots, N\}$. Les positions sont des $(n + 1)$-uplets $c = (j, k_1, \ldots, k_n)$, signifiant que c'est au joueur j de jouer, et qu'il y a k_i allumettes dans la colonne i, avec $0 \leq k_i \leq K$, pour tout i, $1 \leq i \leq n$. Ainsi, $t(j, k_1, \ldots, k_n) = j$. Dans la variante étudiée ici, l'ensemble des coups possibles $\delta(c)$ est défini comme suit :

$$\begin{array}{rcl} \delta(j, k_1, \ldots, k_n) = & \{ & ((j \bmod N) + 1, k'_1, \ldots, k'_n) \\ & | & k'_1, \ldots, k'_n \in \mathbb{N}, \text{ tels que} \\ & & \exists (d_1, \ldots, d_n) \in R \text{ avec } k'_1 + d_1 = k_1 \text{ et } \ldots \text{ et } k'_n + d_n = k_n\} \end{array}$$

où R est l'ensemble des n-uplets suivant, appelé ensemble des règles :

$$\begin{array}{rcl} R & = & \{(u_{in} \bmod 5, u_{in+1} \bmod 5, \ldots, u_{in+n-1} \bmod 5) \\ & & | \ 0 \leq i \leq r - 1\} \setminus \{(0, 0, \ldots, 0)\} \end{array}$$

Question 5 *Lister les règles de R lorsque : **a)** $n = 2, r = 2$; et **b)** $n = 3, r = 3$.*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 6 *Écrire un programme calculant $\delta(c)$ pour chaque configuration c de $\mathcal{C}$, l'ensemble des configurations de $Nim(N, n, K, r)$.*
Que vaut $\delta(c)$ lorsque
a) $N = 2, n = 2, K = 20, r = 2, c = (1, 5, 9)$ *?*
b) $N = 2, n = 3, K = 20, r = 3, c = (2, 3, 7, 2)$ *?*
c) $N = 3, n = 5, K = 20, r = 6, c = (1, 1, 9, 3, 4, 7)$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 7 *Écrire un programme $\mathsf{pre}_\exists$ qui prend en entrée un ensemble $J \subseteq \mathcal{J}$ de joueurs et un ensemble de positions τ, et retourne l'ensemble des positions telles que c'est à un joueur de J de jouer et qu'il existe un coup qui mène dans τ, c'est-à-dire :*

$$\mathsf{pre}_\exists(J, \tau) \quad = \quad \{c \in \mathcal{C} \mid t(c) \in J \text{ et } \exists c' \in \delta(c) \text{ tel que } c' \in \tau\}$$

Que vaut $\mathsf{pre}_\exists(J,\tau)$ *lorsque*
a) $N=2, n=2, K=20, r=2, J=\{1\}, \tau=\{(2,1,5)\}$ *?*
b) $N=2, n=2, K=3, r=2, J=\{1\}, \tau=\{(2,1,1)\}$ *?*
c) $N=2, n=3, K=5, r=3, J=\{1\}, \tau=\{(2,1,2,3),(1,3,2,1)\}$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 8 *Écrire un programme* $\mathsf{pre}_\forall$ *qui prend en entrée un ensemble* $J \subseteq \mathcal{J}$ *de joueurs et un ensemble de positions* τ*, et retourne l'ensemble des positions telles que ce n'est pas à un joueur de* J *de jouer, mais que quel que soit le coup joué, la position suivante est dans* τ *:*

$$\mathsf{pre}_\forall(J,\tau) \;=\; \{c \in \mathcal{C} \mid t(c) \notin J \text{ et } \forall c' \in \delta(c), c' \in \tau\}$$

Combien y a-t-il d'éléments dans $\mathsf{pre}_\forall(J,\tau)$ *lorsque*
a) $N=2, n=2, K=4, r=2, J=\{1\}, \tau=\{(2,0,0)\}$ *?*
b) $N=2, n=2, K=3, r=2, J=\{1\}, \tau=\{(2,1,1)\}$ *?*
c) $N=2, n=3, K=5, r=3, J=\{1\}, \tau=\{(2,0,0,0)\}$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 9 *Écrire un programme* pre *qui prend en entrée un ensemble* $J \subseteq \mathcal{J}$ *de joueurs et un ensemble de positions* τ*, et retourne*

$$\mathsf{pre}(J,\tau) \;=\; \mathsf{pre}_\exists(J,\tau) \cup \mathsf{pre}_\forall(J,\tau)$$

Pour $N=2, n=2, K=20, r=2$*,*
a) *combien y a-t-il de positions* c *dans* $\mathsf{pre}(\{1\}, \{(j,k_1,k_2) \mid j \in \mathcal{J}, k_1 \le 2, k_2 \le 2\})$ *?*
b) *combien y en a-t-il telles que* $t(c)=2$ *?*
Pour $N=3, n=3, K=5, r=3$*, que valent les cardinalités de*
c) $\mathsf{pre}(\{1\}, \{(j,0,0,0) \mid j \in \mathcal{J}\})$ *?*
d) $\mathsf{pre}(\{1,2\}, \{(j,0,0,0) \mid j \in \mathcal{J}\})$ *?*
e) $\mathsf{pre}(\emptyset, \{(j,0,0,0) \mid j \in \mathcal{J}\})$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 10 *Écrire un programme* until *qui prend en entrée un ensemble* $J \subseteq \mathcal{J}$ *de joueurs, deux ensembles* σ *et* τ *de positions, et calcule*

$$\mathsf{until}(J,\sigma,\tau) \;=\; \bigcup_{n\in\mathbb{N}} \mathsf{until}_n(J,\sigma,\tau)$$

où

$$\begin{aligned} \mathsf{until}_0(J,\sigma,\tau) &= \tau \\ \mathsf{until}_{n+1}(J,\sigma,\tau) &= \mathsf{until}_n(J,\sigma,\tau) \cup (\mathsf{pre}(J, \mathsf{until}_n(J,\sigma,\tau)) \cap \sigma) \end{aligned}$$

Indication : on observera que $(\mathsf{until}_n(J,\sigma,\tau))_{n\in\mathbb{N}}$ *forme une suite croissante d'ensembles de positions, pour l'ordre d'inclusion. L'ensemble de toutes les positions étant fini, cette suite est stationnaire : il existe un entier* n *tel que* $\mathsf{until}_k(J,\sigma,\tau) = \mathsf{until}_n(J,\sigma,\tau)$ *pour tout* $k \ge n$*. En particulier,* $\mathsf{until}_n(J,\sigma,\tau) = \mathsf{until}_{n+1}(J,\sigma,\tau)$ *pour cet* n*. Réciproquement, il est facile de*

voir que si $\mathsf{until}_n(J,\sigma,\tau) = \mathsf{until}_{n+1}(J,\sigma,\tau)$ *pour un certain entier* n*, alors* $\mathsf{until}_k(J,\sigma,\tau) = \mathsf{until}_n(J,\sigma,\tau)$ *pour tout* $k \geq n$*.*
Pour $N = 3$*,* $n = 3$*,* $K = 5$*,* $r = 3$*, que valent les cardinalités de*

a) $\mathsf{until}(\{1\}, \{(j,k_1,k_2,k_3) \in \mathcal{C} \mid j \in \mathcal{J}, k_2 \geq k_3\},$
$\{(j,k_1,k_2,k_3) \in \mathcal{C} \mid j \in \mathcal{J}, j \neq 1, \forall (d_1,d_2,d_3) \in R, \exists i, 1 \leq i \leq 3$ *tel que* $d_i > k_i\})$ *?*

b) $\mathsf{until}(\{1,2\}, \{(j,k_1,k_2,k_3) \in \mathcal{C} \mid j \in \mathcal{J},\},$
$\{(j,k_1,k_2,k_3) \in \mathcal{C} \mid j \in \mathcal{J}, j \neq 2, \forall (d_1,d_2,d_3) \in R, \exists i, 1 \leq i \leq 3$ *tel que* $d_i > k_i\})$ *?*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

4 Logique du temps alternant

Étant donné un jeu $(\mathcal{J}, \mathcal{C}, t, \delta)$, on définit des formules F comme suit. La logique résultante s'appelle ATL ("alternating time logic"). Ces formules spécifient des propriétés de positions du jeu, et on définira ce que veut dire que F est vraie à la position $c \in \mathcal{C}$, en notation $c \models F$, en même temps que l'on définira les formules F elles-mêmes.

- Le symbole $\mathbf{tour}_J$ est une formule pour tout $J \subseteq \mathcal{J}$. On a $c \models \mathbf{tour}_J$ si et seulement si $t(c) \in J$ — c'est au tour d'un joueur de J de jouer.
- Si F et G sont deux formules, alors $F \wedge G$, $F \vee G$, $\neg F$, $F \Rightarrow G$, $\top$, $\bot$ sont des formules. On a $c \models F \wedge G$ ("F et G") si et seulement si $c \models F$ et $c \models G$; $c \models F \vee G$ ("F ou G") si et seulement si $c \models F$ ou $c \models G$; $c \models \neg F$ ("non F") si et seulement si $c \not\models F$; $c \models F \Rightarrow G$ ("si F alors G") si et seulement si $c \not\models F$ ou $c \models G$; $c \models \top$ toujours ("vrai") ; $c \models \bot$ jamais ("faux").
- Si F est une formule et $J \subseteq \mathcal{J}$ un ensemble de joueurs, alors $\langle\langle J \rangle\rangle \bigcirc F$ est une formule. On a $c \models \langle\langle J \rangle\rangle \bigcirc F$ si et seulement si, informellement, les joueurs de J peuvent former une coalition, de sorte que quoi que fassent les autres joueurs, la position atteinte après un coup vérifie F.
  Formellement, $c \models \langle\langle J \rangle\rangle \bigcirc F$ si et seulement si $t(c) \in J$ et il existe un coup que $t(c)$ peut jouer, c'est-à-dire il existe $c' \in \delta(c)$, tel que $c' \models F$, ou bien $t(c) \notin J$ et quoi que $t(c)$ joue, c'est-à-dire pour tout $c' \in \delta(c)$ on a $c' \models F$.
- Si F et G sont deux formules et $J \subseteq \mathcal{J}$ un ensemble de joueurs, alors $\langle\langle J \rangle\rangle F$ Until G est une formule. Informellement, $\langle\langle J \rangle\rangle F$ Until G est vraie si et seulement si les joueurs de J peuvent se coaliser pour que, quoi que fassent les autres joueurs, on finisse par arriver en un nombre fini arbitraire de coups à une position où G est vraie, et en attendant, F est restée vraie à toutes les positions intermédiaires.
  Formellement, $c \models \langle\langle J \rangle\rangle F$ Until G est vraie si et seulement si il existe un entier $n \in \mathbb{N}$ tel que $c \models \langle\langle J \rangle\rangle F$ Until_n G, où la formule $\langle\langle J \rangle\rangle F$ Until_n G est définie par

$$\begin{aligned} \langle\langle J \rangle\rangle F \ \mathsf{Until}_0 \ G &= G \\ \langle\langle J \rangle\rangle F \ \mathsf{Until}_{n+1} \ G &= (\langle\langle J \rangle\rangle F \ \mathsf{Until}_n \ G) \vee (F \wedge (\langle\langle J \rangle\rangle \bigcirc (\langle\langle J \rangle\rangle F \ \mathsf{Until}_n \ G))) \end{aligned}$$

Par exemple, la formule $\mathbf{tour}_J \wedge \langle\langle J \rangle\rangle \bigcirc \top$ signifie informellement qu'il y a un joueur de J qui peut jouer tout de suite (c'est son tour, $\mathbf{tour}_J$, et les joueurs de J peuvent former une coalition de sorte à pouvoir jouer un coup).

Encore à titre d'exemple, la formule $\langle\langle\emptyset\rangle\rangle\top$ Until $\neg\mathbf{tour}_J$ signifie que quoi que l'on fasse, au bout d'un moment ce ne sera plus le tour de jouer d'aucun joueur de J (quoi que fassent les joueurs, au bout d'un moment $\mathbf{tour}_J$ deviendra faux).
La formule $\langle\langle\emptyset\rangle\rangle\top$ Until $(\mathbf{tour}_J \wedge (\langle\langle J\rangle\rangle \bigcirc \top))$ signifie que quoi qu'il arrive, au bout d'un moment un joueur de J pourra jouer : en effet, quoi que fassent les joueurs, au bout d'un moment la formule $\mathbf{tour}_J \wedge \langle\langle J\rangle\rangle \bigcirc \top$ vue plus haut sera vraie.
Notre dernier exemple sera la formule $\mathbf{tour}_{\mathcal{J}\setminus J} \wedge (\langle\langle J\rangle\rangle \bigcirc \bot)$, où $\setminus$ désigne la différence ensembliste. Elle exprime que c'est le tour d'un opposant, c'est-à-dire un joueur hors de la coalition J, mais que tous les opposants sont bloqués. En effet, $\langle\langle J\rangle\rangle \bigcirc \bot$ signifie que les joueurs de J peuvent les empêcher de jouer.

Question 11 *Écrire un programme prenant en entrée quatre entiers N, n, K, r, ainsi qu'une formule F, et retourne l'ensemble des positions c du jeu $Nim(N,n,K,r)$ telles que $c \models F$. On pourra s'aider des fonctions* pre *et* until *de la partie 3.*
Soit F_J la formule

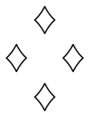
$$\langle\langle J\rangle\rangle(\langle\langle \mathcal{J}\rangle\rangle \bigcirc \top) \text{ Until } (\mathbf{tour}_{\mathcal{J}\setminus J} \wedge (\langle\langle J\rangle\rangle \bigcirc \bot))$$

qui exprime que les joueurs de J peuvent se coaliser de sorte à faire perdre un joueur de $\mathcal{J} \setminus J$, après un nombre fini indéterminé de coups.
Dans le jeu $Nim(N,n,K,r)$, on dira qu'une position initiale est une position de la forme $(j,k_1,\ldots,k_n)$ avec $k_1 = k_2 = \ldots = k_n = k$, où $k \leq K$. Pour $N=3$, $n=2$, $K=20$, $r=3$,
a) *Quelles sont les positions initiales où le joueur 1 est sûr de gagner ? (les c initiaux tels que $c \models F_{\{1\}}$?)*
b) *Combien y a-t-il de positions c à partir desquelles 1 est sûr de gagner (combien y a-t-il de c tels que $c \models F_{\{1\}}$?)*
c) *Combien y a-t-il de positions c à partir desquelles 1 et 2 peuvent s'allier pour être sûr que 3 perde (combien y a-t-il de c tels que $c \models F_{\{1,2\}}$?)*
d) *Parmi celles-ci, combien sont initiales ?*
★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 12 *On va essayer de déterminer s'il vaut mieux jouer tout seul, jouer en collaborant avec d'autres joueurs, ou bien d'abord collaborer avec d'autres pour mieux les trahir ensuite. On définit les formules :*

$$\begin{aligned} Gagne_j &= \langle\langle\{j\}\rangle\rangle(\langle\langle \mathcal{J}\rangle\rangle \bigcirc \top) \text{ Until } (\mathbf{tour}_{\mathcal{J}\setminus\{j\}} \wedge \langle\langle\{j\}\rangle\rangle \bigcirc \bot) \\ Perd_j &= \neg Gagne_j \\ Traitre_1 &= \langle\langle\{1,2\}\rangle\rangle(\langle\langle \mathcal{J}\rangle\rangle \bigcirc \top) \text{ Until } (Perd_3 \wedge Gagne_1) \end{aligned}$$

où $Gagne_j$ exprime que le joueur j peut jouer seul et gagner (en faisant perdre un autre joueur), $Perd_j$ exprime que le joueur j est à une position où il ne peut plus gagner seul, et $Traitre_1$ exprime que le joueur 1 peut gagner en collaborant d'abord avec 2 jusqu'au moment où l'on est sûr que 3 va perdre s'il joue seul, à partir duquel 1 rompt son alliance avec 2 et joue seul.
a) *Combien y a-t-il de positions c telles que $c \models Traitre_1$ pour $N=3$, $n=2$, $K=20$, $r=3$?*
b) *Combien de ces positions sont-elles initiales ?*



Jeux à plusieurs et coalitions

Nom, prénom, u_0: . . . . . .

Question 1

a)

b)

c)

d)

Question 2

Question 3

a)

b)

c)

Question 4

a)

b)

c)

d)

Question 5

a)

b)

Question 6

a)

b)

c)

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

c)

d)

e)

Question 10

a)

b)

Question 11

a)

b)

c)

d)

Question 12

a)

b)