AAGA Examen Réparti 1

Appareils électroniques interdits. Seuls documents autorisés : Notes de cours et de TD/TME.

Le sujet comporte 2 exercices indépendants. Le barème est indicatif. L'examen est noté sur 20, il y a donc 4 points en bonus.

Exercice 1 *: Générateur à la Fibonacci (10 points)*

On appelle *générateur de Fibonacci décalé*, un générateur du type :

$$X_n = X_{n-r} \oplus X_{n-s}, \qquad \text{avec des entiers } 0 < r < s.$$

Attention l'addition $\oplus$ est une addition binaire bit-à-bit sans retenue. Ainsi on a par exemple $6 \oplus 3 = 5$ car $6 =_2 110$ et $3 =_2 11$, en écrivant les bits de poids faible à droite. L'addition des deux bits de poids faible donne 1, les deux bits suivants sont 1 et 1 dont l'addition donne 0 (on ne conserve pas la retenue) et le bit de poids fort 1 est additionné à 0. On peut voir l'opérateur $\oplus$ comme le *ou exclusif bit-à-bit*. Le générateur a besoin de s graines $X_0, X_1, \ldots, X_{s-1}$ pour être bien défini.

Générateur particulier

On choisit $r = 3$ et $s = 4$ et les graines $X_0 = 4, X_1 = 19, X_2 = 1$ et $X_3 = 5$.

Question 1 Sans effectuer de calcul, les graines fournies nous informent sur une borne supérieure des valeurs prises par le générateur. Quelle est cette borne maximale ? (justifier la réponse)

Question 2 Générer tous les entiers jusqu'à X_{10} inclus, en justifiant les réponses par l'écriture en binaire des entiers.

Question 3 A-t-on trouvé une période à partir des entiers X_0 à X_{10} ?

Question 4 Pour quelle raison pour ce générateur, alors que $X_0 = X_6$, il n'est pas vrai que $X_1 = X_7$?

Question 5 Donner une borne supérieure sur la longueur de la période de ce générateur. Justifier la réponse.

Générateur quelconque

Question 6 Prouver que ce type de générateurs est périodique (r et s étant quelconques), quelque soit l'ensemble de s graines fourni.

Soit μ et λ les deux plus petits entiers tels que pour tout $n \geq \mu$, on a $X_{n+\lambda} = X_n$.
Les entiers μ et λ existent d'après la question précédente.

Question 7 Montrer qu'il existe un entier $n \geq \mu$ tel que les s entiers consécutifs $X_n, X_{n+1}, \ldots, X_{n+s-1}$ sont respectivement égaux aux s entiers $X_{2n}, X_{2n+1}, \ldots, X_{2n+s-1}$.
On suppose que le plus petit entier $n_0 > 0$ vérifiant la question précédente vérifie $\mu \leq n_0 \leq \mu + \lambda$.

Question 8 Donner le pseudo-code d'une fonction prenant en paramètres les entiers r, s et une liste de s graines et renvoyant les valeurs μ et λ en seulement $O(\mu + \lambda)$ additions et avec un espace mémoire $O(s)$.

Exercice 2 : Génération uniforme de tas (14 points)

Dans tout l'exercice, les tableaux sont indexés à partir de 1.

Unranking de combinaison

On définit une combinaison de k éléments parmi n comme un tableau indexé de 1 à k, et contenant k entiers distincts entre 1 et n, rangés de façon croissante.
On rappelle que le nombre de combinaisons de k éléments parmi n est égal au coefficient binomial $\binom{n}{k}$.

Voilà un résultat que nous allons exploiter par la suite.
Soit $n \geq k > 0$. Pour tout r, vérifiant $0 \leq r < \binom{n}{k}$, il existe une unique suite $0 \leq c_1 < c_2 < \cdots < c_k < n$ de telle sorte que

$$r = \binom{c_1}{1} + \binom{c_2}{2} + \cdots + \binom{c_k}{k}.$$

Par exemple, pour $n = 5$ et $k = 3$, l'entier 8 est égal à $\binom{1}{1} + \binom{3}{2} + \binom{4}{3}$.
Afin de calculer la suite $(c_1, \ldots, c_k)$ associée à r (et n et k), on procède ainsi :

(i) c_k est l'entier le plus grand tel que $\binom{c_k}{k} \leq r$;

(ii) on remarque que le reste $r - \binom{c_k}{k}$ est inférieur (strictement) à $\binom{n-1}{k-1}$, et on appelle récursivement l'algorithme sur $r - \binom{c_k}{k}$, $n-1$ et $k-1$.

Question 1 Définir une fonction récursive `combinadic` prenant les 3 entiers r, n et k en paramètres et renvoyant un tableau contenant la suite $(c_1, \ldots, c_k)$. Ne pas oublier de traiter le cas de base (omis dans l'explication précédente...).
Si vous faites appel à des primitives, il faut les décrire (signature + description).

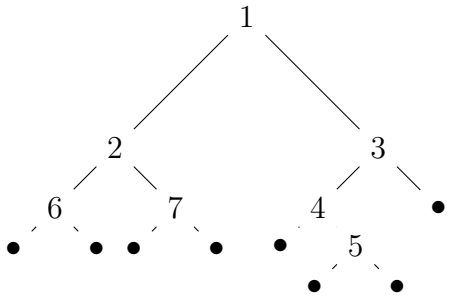
On a quasiment un algorithme de unranking de combinaisons. Il suffit d'associer au rang r les entiers $(c_1 + 1, c_2 + 1, \ldots, c_k + 1)$

Question 2 Définir une fonction `combi_unrank` de unranking de la combinaison de rang r associée aux entiers n et k, en s'appuyant sur la remarque précédente.

Question 3 La méthode de unranking renvoie-t-elle les combinaisons dans l'ordre lexicographique ? (justifier la réponse)

Générateur d'arbres binaires croissants

Un arbre binaire est une structure arborescente décomposable de la manière suivante. Un arbre binaire est soit (1) une feuille •, soit (2) un nœud interne ◦ avec un enfant gauche et un enfant droit qui sont chacun des arbres binaires.
Un arbre binaire croissant est un arbre binaire, dont les nœuds internes sont étiquetés avec un entier entre 1 et n (qui est le nombre total de nœuds internes), chaque entier apparaissant exactement une fois, et tel que les suites d'étiquettes de la racine vers n'importe quelle feuille est une suite strictement croissante.
Voilà un exemple de taille 7 :



Question 4 Construire les 6 arbres de taille 3.

La suite de comptage des arbres croissants est donnée par :

$$\begin{aligned} A_0 &= 1 \\ A_n &= \sum_{k=0}^{n-1} \binom{n-1}{k} A_k \, A_{n-1-k} \qquad \texttt{si } n > 0. \end{aligned}$$

Question 5 Donner une interprétation combinatoire de cette équation.

Question 6 Calculer les premières valeurs de la suite A_n pour n allant de 0 jusqu'à 5.

Question 7 Prouver que $A_n = n!$ pour tout $n \in \mathbb{N}$.

Question 8 Décrire un ordre total sur les arbres croissants de taille n.

Question 9 Donner un algorithme `gen_rec_arbre(`n`)` de type générateur récursif qui construit uniformément un arbre de taille n.
On suppose que la valeurs de tous les $x!$, pour $x \in \{0, \ldots, n\}$, ont été précalculées et sont accessibles en temps $O(1)$.
Ne pas oublier l'étiquetage des nœuds internes !

Question 10 Donner un algorithme `unrank_arbre(`r, n`)` de type unranking qui construit l'arbre de taille n et de rang r (selon l'ordre défini au dessus).
On suppose que la valeurs de tous les $x!$, pour $x \in \{0, \ldots, n\}$ ont été précalculées et sont accessibles en temps $O(1)$.
Ne pas oublier l'étiquetage des nœuds internes !

Question 11 Quelle est la mesure de complexité temporelle ayant le plus de sens pour l'algorithme `gen_rec_arbre(`n`)` ? Calculer la complexité temporelle de `gen_rec_arbre(`n`)`, lorsque n tend vers l'infini.

Question subsidiaire

Question 12 Montrer que la fonction `combi_unrank(`r, n, k`)` est correcte.