TP / PC*

Une chasse au trésor

Ce sujet s'intéresse à une chasse au trésor sur des graphes. Il vous donne l'occasion de réviser les notions que vous avez apprises à leur sujet : représentation par listes d'adjacence, parcours d'un graphe, recherche de plus court chemin.

En cas de difficulté, vous pouvez me contacter par l'intermédiaire de mon site : `https://pc-etoile.schola.fr/`

1. Génération pseudo-aléatoire d'une suite d'entiers

On définit une suite (u_k) par la donnée de la condition initiale $u_0 = 42$ et la relation de récurrence

$$\forall k \in \mathbb{N}, \quad u_{k+1} = 19\,999\,999 \times u_k \bmod (19\,999\,981).$$

Cette suite sera utilisée pour générer des graphes pseudo-aléatoires. Pour que la création de ces derniers ne soit pas trop longue, il est indispensable de stocker dans une liste les différentes valeurs de u_k, pour $0 \leqslant k < 1\,000\,000$.

Question 1. Définir et remplir cette liste, puis donner les valeurs suivantes :

a) $u_{123} \bmod (1\,000)$ b) $u_{456} \bmod (1\,000)$ c) $u_{789} \bmod (1\,000)$

2. Génération d'un graphe

À partir de la suite (u_k) et de deux paramètres entiers $n \in \mathbb{N}^*$ et $p \in \mathbb{N}$ on construit un graphe non orienté $G = (S, A)$. Les sommets S de G sont les entiers de 0 à $n-1$: $S = \{0, 1, 2, \dots, n\}$ et on définit les arêtes A de G en s'aidant de la suite (v_k) définie par :

$$\forall k \in \mathbb{N}, \qquad v_k = \begin{cases} 1 & \text{si } u_k \bmod (10\,000) < p \\ 0 & \text{sinon} \end{cases}$$

Le graphe s'obtient en considérant que les $n-1$ premières valeurs de (v_k) sont caractéristiques des arêtes reliant le sommet 0 aux autres sommets du graphe :

- $\{0, 1\} \in A$ si et seulement si $v_0 = 1$;
- $\{0, 2\} \in A$ si et seulement si $v_1 = 1$;
- ...
- $\{0, n-1\} \in A$ si et seulement si $v_{n-2} = 1$.

Les $n-2$ valeurs suivantes sont caractéristiques des arêtes reliant le sommet 1 aux sommets d'indices supérieurs :

- $\{1, 2\} \in A$ si et seulement si $v_{n-1} = 1$;
- $\{1, 3\} \in A$ si et seulement si $v_n = 1$;
- ...
- $\{1, n-1\} \in A$ si et seulement si $v_{2n-4} = 1$.

et ainsi de suite jusqu'à :

- $\{n-3, n-2\} \in A$ si et seulement si $v_{n(n-1)/2-3} = 1$;
- $\{n-3, n-1\} \in A$ si et seulement si $v_{n(n-1)/2-2} = 1$;
- $\{n-2, n-1\} \in A$ si et seulement si $v_{n(n-1)/2-1} = 1$.

Question 2. Calculer le nombre d'arêtes du graphe $G = (S, A)$ pour les valeurs de n et de p suivantes :

a) $n = 10$ et $p = 654$ b) $n = 100$ et $p = 543$ c) $n = 1\,000$ et $p = 12$

Les valeurs obtenues à la question 2 montrent que pour les valeurs de p que nous allons utiliser il est plus économique de représenter en mémoire le graphe $G = (S, A)$ par ses listes d'adjacence : ainsi, le graphe G est défini par une liste de listes `a = [a[0], a[1], ..., a[n-1]]` où `a[k]` est la liste des sommets accessibles à partir du sommet k.

Important. Rappelons que G est un graphe *non orienté* ; ainsi, si $j \in a[i]$, alors $i \in a[j]$.

Question 3. Calculer le nombre de sommets dans la composante connexe du sommet 0, c'est-à-dire le nombres de sommets qui sont reliés à 0 par un chemin, pour les valeurs de n et p suivantes :

a) $n = 10$ et $p = 654$ b) $n = 100$ et $p = 543$ c) $n = 1\,000$ et $p = 12$

Il sera judicieux de commencer par définir une fonction qui prend pour arguments n et p et renvoie le graphe G défini par ses listes d'adjacence.

Question 4. Calculer le nombre de composantes connexes de G pour les valeurs de n et p suivantes :

a) $n = 10$ et $p = 654$ b) $n = 100$ et $p = 543$ c) $n = 1\,000$ et $p = 12$

3. Une chasse au trésor

Afin de s'assurer que les graphes soient bien connexes, nous ajouterons à présent à tous les graphes générés les arêtes de la forme $\{i, i+1\}$ pour $0 \leqslant i < n-1$.

La *distance* entre deux sommets est la longueur du plus court chemin entre eux, la longueur d'un chemin étant égal au nombre d'arêtes qui le composent.

On fixe un nombre k de pièces d'or sur les sommets $n-k$, $n-k+1, \ldots, n-2$, $n-1$ et l'on part du sommet 0. Le but est de ramasser toutes les pièces d'or en cherchant à minimiser la distance totale parcourue. Pour ce faire, on propose la stratégie gloutonne suivante : à chaque instant on se déplace vers la pièce d'or la plus proche. Dans le cas où deux pièces d'or seraient à même distance, on choisit celle située sur le sommet de plus petit indice.

Question 5. Déterminer le nombre de mouvements de la stratégie gloutonne pour ramasser toutes les pièces d'or pour les valeurs suivantes :

a) $n = 100$, $p = 32$ et $k = 10$ b) $n = 200$, $p = 21$ et $k = 15$ c) $n = 1\,000$, $p = 1$ et $k = 20$

Question 6. Donner un exemple simple de graphe pour lequel la stratégie gloutonne ne donne pas un résultat optimal.

Question 1. $u_k \bmod 1000$: 768 pour $k = 123$ 761 pour $k = 456$ 593 pour $k = 789$

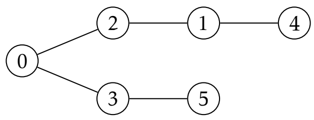
Question 2. Nombre d'arêtes : 3 pour $(n,p) = (10,654)$ 261 pour $(n,p) = (100,543)$ 607 pour $(n,p) = (1\,000,12)$

Question 3. Nombre de sommets : 2 pour $(n,p) = (10,654)$ 100 pour $(n,p) = (100,543)$ 1 pour $(n,p) = (1\,000,12)$

Question 4. Nombre de composantes : 7 pour $(n,p) = (10,654)$ 1 pour $(n,p) = (100,543)$ 394 pour $(n,p) = (1\,000,12)$

Question 5. Distance parcourue : 27 pour $(n,p,k) = (100,32,10)$ 35 pour $(n,p,k) = (200,21,15)$ 36 pour $(n,p,k) = (1\,000,1,20)$

Question 6. Un exemple pour $n = 6$ et $k = 4$ (les pièces sont sur les sommets 2, 3, 4 et 5) :



L'algorithme glouton fait suivre un chemin de longueur 9 : 0 – 2 – 0 – 3 – 5 – 3 – 0 – 2 – 1 – 4.
Il existe un chemin de longueur 7 : 0 – 3 – 5 – 3 – 0 – 2 – 1 – 4.