

Automates pour rechercher

Épreuve pratique d'algorithmique et de programmation

Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin/juillet 2007

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 Introduction

La recherche d'un mot dans un texte est un problème classique. Les éditeurs de texte offrent tous une telle fonctionnalité. La difficulté est de pouvoir le faire de façon efficace sur de longs textes. En effet l'ergonomie d'un éditeur de texte tient notamment au fait que la recherche d'un mot puisse se faire en un temps à peine perceptible par l'utilisateur. C'est pourquoi il est nécessaire d'employer des algorithmes efficaces, par exemple, en utilisant des automates.

2 Mots pseudo-aléatoires

Considérons la suite d'entiers (u_n) définie pour $n \geq 0$ par :

$$u_n = \begin{cases} \text{votre } u_0 \text{ (à reporter sur votre fiche)} & \text{si } n = 0 \\ 15\,091 \times u_{n-1} \pmod{64\,007} & \text{si } n \geq 1 \end{cases}$$

Soit m un entier positif non-nul. La suite d'entiers $(v_{m,n})$ est définie pour $n \geq 0$ par :

$$v_{m,n} = u_n \pmod{m}$$

Question 1 **a)** Quelle est la valeur de $v_{7,1000}$? **b)** Quelle est la valeur de $v_{10,10\,000}$? **c)** Quelle est la valeur de $v_{3,100\,000}$?

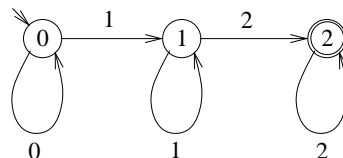
On appelle mot toute suite finie de lettres. Le mot de longueur nulle est noté ϵ . On note $x.y = x_1 \dots x_p y_1 \dots y_q$ la concaténation de deux mots $x = x_1 \dots x_p$ et $y = y_1 \dots y_q$.

On note $w_{n,k,l}$ le mot constitué des lettres $v_{n,k} \dots v_{n,k+l-1}$.

3 Automates

Notons E_n l'ensemble d'entiers naturels $\{0, \dots, n-1\}$. On appelle automate sur l'alphabet E_n , tout quadruplet $\mathcal{A} = (Q, I, F, T)$, où Q est un ensemble fini d'états, $I \subseteq Q$ est un ensemble d'états initiaux, $F \subseteq Q$ est un ensemble d'états finals et $T \subseteq Q \times E_n \times Q$ une relation de transition.

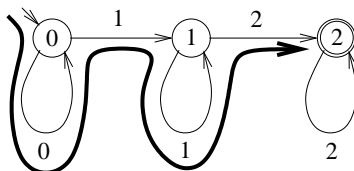
Les automates sont souvent représentés par des graphes dirigés dont les sommets sont les états de l'automate et dont les arcs représentent la relation de transition. Les états initiaux sont désignés par une petite flèche entrante et les états finals sont désignés par un double cercle. La figure ci-dessous donne un exemple d'automate :



Un chemin Γ de l'automate est une suite alternée états-lettres $\Gamma = q_0, e_1, q_1, e_2, q_2, \dots, e_l, q_l$ telle que $\forall i \in \{0, \dots, l-1\}, (q_i, e_{i+1}, q_{i+1}) \in T$.

Le mot formé par le chemin Γ est le mot formé des lettres $e_1 e_2 \dots e_n$. Un chemin $\Gamma = q_0, e_1, q_1, e_2, q_2, \dots, e_l, q_l$ est dit accepté par l'automate \mathcal{A} si l'état de départ est initial, $q_0 \in I$ et l'état d'arrivée est final, $q_l \in F$. Un mot $w = e_1 \dots e_l$ est reconnu par l'automate \mathcal{A} , si et seulement si il existe un chemin $\Gamma = q_0, e_1, q_1, e_2, q_2, \dots, e_l, q_l$ accepté par \mathcal{A} . Le langage de l'automate \mathcal{A} est l'ensemble des mots reconnus par \mathcal{A} .

Ainsi, l'automate de la figure ci-dessus reconnaît le mot 0112, puisque que le chemin ci-dessous va de l'unique état initial vers l'unique état final.

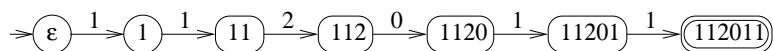


L'automate \mathcal{A} est dit déterministe si et seulement si I est un singleton et T est une fonction de $Q \times E_n$ dans Q , c'est à dire que pour tout état $q \in Q$ et toute lettre $e \in E_n$ il existe au plus un $q' \in Q$ tel que (q, e, q') appartienne à T .

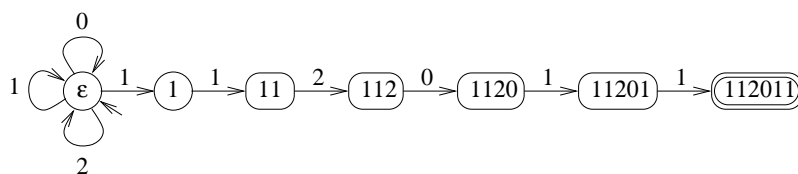
4 Recherche d'un mot dans un texte

Nous souhaitons rechercher toutes les occurrences d'un mot $y = f_1 \dots f_k$ formé de lettres appartenant à E_n dans un mot $z = e_1 \dots e_l$ lui aussi formé de lettres appartenant à E_n . Il s'agit donc d'énumérer les mots $t = e_1 \dots e_m$, $m \leq l$, préfixes de z et terminant par le mot y . Cette recherche peut se faire efficacement sur des mots de grande taille, à l'aide d'un automate.

Prenons un exemple. Pour rechercher rapidement le mot $y = 112011$, on peut considérer l'automate \mathcal{A}_y donné ci-dessous, dont le langage est le singleton $\{y\}$.

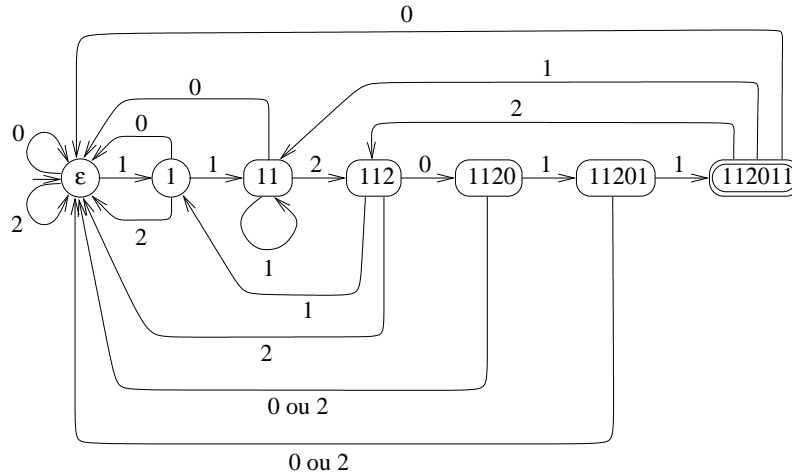


Considerons maintenant l'automate \mathcal{A}'_y donné ci-dessous.



Il s'agit simplement de l'automate \mathcal{A}_y auquel on a ajouté, pour chaque lettre e de l'alphabet E_n , une transition (ϵ, e, ϵ) bouclant sur l'état initial. Ce nouvel automate reconnaît le langage des mots terminant par y . Malheureusement, il est non-déterministe, et ne peut être utilisé directement pour résoudre le problème de recherche. Il faut le transformer en automate déterministe.

La particularité de l'automate \mathcal{A}'_y est qu'il est très facile de construire un automate déterministe reconnaissant le même langage. Pour cela, il suffit de prendre l'automate \mathcal{A}_y , et de le compléter, par ajout de transitions, en l'automate \mathcal{A}''_z , donné ci-dessous :



Les automates \mathcal{A}'_y et \mathcal{A}''_y reconnaissent le même langage, et \mathcal{A}''_y est déterministe. C'est ce dernier qui peut être utilisé pour rechercher les occurrences de y .

Question 2 Calculer le nombre d'occurrences du mot $y = 112011$ dans les mots $w_{3,0,l}$, pour les valeurs suivantes de l : **a)** $l = 1\ 000$ **b)** $l = 10\ 000$ **c)** $l = 100\ 000$

Nous allons maintenant nous intéresser à la construction de la relation de transition de l'automate \mathcal{A}''_y , pour un mot y quelconque. Pour cela, il est utile de définir le bord d'un mot z comme étant le plus grand mot $\delta(z)$ qui est à la fois préfixe et suffixe de z , mais aussi préfixe de y . Ainsi, le bord de $y = 112011$ est le mot $\delta(y) = 11$.

La relation de transition T de \mathcal{A}''_y est la plus petite relation telle que :

- $(\epsilon, e, e) \in T$, si e est la première lettre de y
- $(\epsilon, e, \epsilon) \in T$, pour toute lettre e différente de la première lettre de y

Pour tout mot z , préfixe de longueur non nulle du mot y :

- $(z, e, z.e) \in T$, si $z.e$ est préfixe du mot y
- $(z, e, \delta(z.e)) \in T$, si $z.e$ n'est pas préfixe du mot y

Question à développer pendant l'oral : Détailler les structures de données employées pour construire l'automate \mathcal{A}''_y . Quelle est la complexité en temps et en mémoire de cette construction, en fonction du cardinal de l'alphabet et de la longueur du mot y ?

Question 3 a) Quel est le nombre d'occurrences du mot $w_{5,0,10}$ dans le mot $w_{5,10,1\ 000}$? **b)** Quel est le nombre d'occurrences du mot $w_{7,0,30}$ dans le mot $w_{5,30,10\ 000}$? **c)** Quel est le nombre d'occurrences du mot $w_{10,0,100}$ dans le mot $w_{5,100,100\ 000}$?

On modifiera l'algorithme pour calculer le plus grand préfixe du mot y ayant au moins une occurrence dans un texte donné.

Question 4 a) Quelle est la longueur du plus grand préfixe du mot $w_{5,0,100}$ ayant au moins une occurrence dans le mot $w_{5,100,10\ 000}$? **b)** Quelle est la longueur du plus grand préfixe du mot $w_{7,0,300}$ ayant au moins une occurrence dans le mot $w_{7,300,30\ 000}$? **c)** Quelle est la longueur du plus grand préfixe du mot $w_{10,0,1\ 000}$ ayant au moins une occurrence dans le mot $w_{10,1\ 000,100\ 000}$?

Modifier à nouveau l'algorithme pour compter le nombre d'occurrences du plus grand préfixe d'un mot y ayant au moins une occurrence dans un texte.

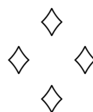
Question 5 **a)** *Quel est le nombre d'occurrences du plus grand préfixe du mot $w_{5,0,100}$ ayant au moins une occurrence dans le mot $w_{5,10,10\,000}$?* **b)** *Quel est le nombre d'occurrences du plus grand préfixe du mot $w_{7,0,300}$ ayant au moins une occurrence dans le mot $w_{7,300,30\,000}$?* **c)** *Quel est le nombre d'occurrences du plus grand préfixe du mot $w_{10,0,1\,000}$ ayant au moins une occurrence dans le mot $w_{10,1\,000,100\,000}$?*

Question à développer pendant l'oral : Détailler l'algorithme utilisé pour calculer la longueur du plus grand préfixe du mot y ayant au moins une occurrence dans un texte. Quelle est la complexité en temps de cette recherche, en fonction de la longueur du texte ? Modifier l'automate construit pour pouvoir compter le nombre d'occurrences de mots ne différant d'un mot y donné que par l'altération d'une lettre $e \in E_n$ quelconque. On recherche donc les occurrences dans un texte des mots $z = y'.e.y''$, avec $y = y'.f.y''$, et $e, f \in E_n$. Pour produire un automate déterministe, il faudra adapter la méthode décrite précédemment.

Question 6 **a)** *Quel est le nombre d'occurrences, à l'altération d'une lettre près, du mot $w_{5,0,100}$ dans le mot $w_{5,100,10\,000}$?* **b)** *Quel est le nombre d'occurrences, à l'altération d'une lettre près, du mot $w_{7,0,300}$ dans le mot $w_{7,300,30\,000}$?* **c)** *Quel est le nombre d'occurrences, à l'altération d'une lettre près, du mot $w_{10,0,1\,000}$ dans le mot $w_{10,1\,000,100\,000}$?*

Question à développer pendant l'oral : Détailler la construction de l'automate déterministe reconnaissant les occurrences d'un mot, à l'altération d'une lettre près. Modifier l'automate construit pour compter le nombre d'occurrences de mots ne différant d'un mot y donné que par l'insertion, l'altération ou l'élimination d'une lettre. On recherche donc les occurrences des mots $z = y'.e.y''$, avec $y = y'.f.y''$, et $e, f \in E_n \cup \{\epsilon\}$.

Question 7 **a)** *Quel est le nombre d'occurrences, à une lettre près, du mot $w_{5,0,100}$ dans le mot $w_{5,100,10\,000}$?* **b)** *Quel est le nombre d'occurrences, à une lettre près, du mot $w_{7,0,300}$ dans le mot $w_{7,300,30\,000}$?* **c)** *Quel est le nombre d'occurrences, à une lettre près, du mot $w_{10,0,1\,000}$ dans le mot $w_{10,1\,000,100\,000}$?*



Automates pour rechercher

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)



Fiche d'évaluation: Automates pour rechercher

Nom, prénom, u_0 :

Question 1 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 2 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 3 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 4 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

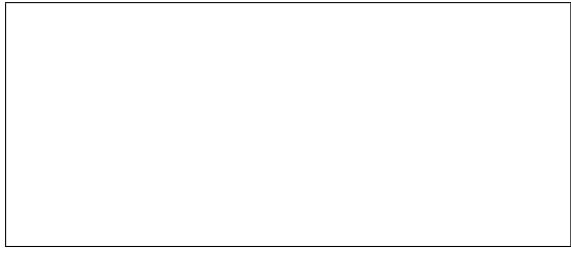
Question 5 0-1-2-3-4-5-6-7-8

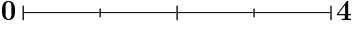
- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

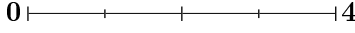
Question 6 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

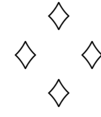
Question 7 0-1-2-3-4-5-6-7-8



a) 0  4

b) 0  4

c) 0  4



La Guerre des Étoiles

Épreuve pratique d'algorithmique et de programmation

Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin/juillet 2007

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*. 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 Introduction

Il y a bien longtemps dans une galaxie lointaine, très lointaine, les humains des forces de l'alliance ont développé une technologie leur permettant d'emporter une victoire définitive contre les races extra-terrestres. Cette nouvelle technologie leur a permis de produire d'énormes vaisseaux de combat connus sous le nom de "Dents de Sabre" et de puissance équivalente aux redoutables bases de défense alien : les bases "Mammouths". À cette époque, les forces de l'alliance contrôlaient un certain nombre de planètes alors que d'autres étaient sous le contrôle des aliens. À l'aide des "Dents de Sabre", les humains ont fini par emporter la victoire contre les aliens, à l'issue de ce qui fut la première guerre interplanétaire de l'histoire. Notre objectif est d'effectuer quelques simulations afin de vérifier certaines hypothèses historiques.

La production de chaque vaisseau nécessite une certaine quantité de temps constante au cours du temps et propre à chaque planète. Le taux de production de la planète i est noté ρ_i (en nombre de vaisseaux par année) et chaque planète dispose initialement d'un certain nombre n_i de vaisseaux. Chaque planète commence à produire des vaisseaux dès le début de la simulation. Ainsi, au temps t , la planète i dispose de $n_i + \rho_i \cdot t$ vaisseaux. La défense des planètes alien est assurées par les puissantes bases Mammouths. De même, chaque planète alien j dispose initialement d'un certain nombre n_j de bases et en produit ρ_j par an.

Lors d'un affrontement entre les vaisseaux de l'alliance et les bases alien, l'armée en supériorité numérique emporte le combat. En cas de victoire de l'alliance, cette dernière prend le contrôle de la planète. En cas d'égalité des forces en présences, c'est l'alliance qui emporte la victoire.

Bol, le général en chef de l'alliance avait décidé de la stratégie suivante : pour chaque planète alien j , il choisit une planète de l'alliance i qui produira des vaisseaux jusqu'à la date t_i et qui seront tous envoyés vers la planète j . Une planète alien ne peut donc être envahie que par au plus une planète de l'alliance et aucune planète de l'alliance n'envoie ses vaisseaux vers deux planètes alien différentes. Enfin, Bol avait également décidé que pour profiter de l'effet de surprise tous les vaisseaux des forces de l'alliance partiraient à la même date.

Une des difficulté tient au fait qu'il faut un temps $\delta_{i,j}$ (parfois infini en raison de l'inexistence de voie inter-galactique) pour se rendre de la planète i à la planète j . Et pendant ce temps, les planètes alien continuent de produire leurs bases de défense.

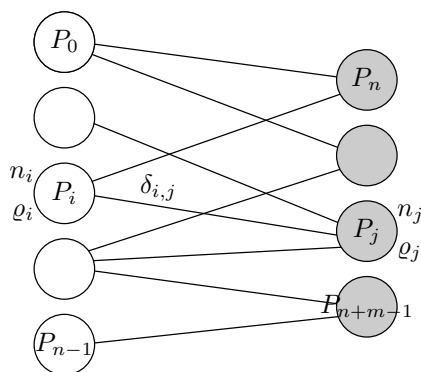
Notre objectif est d'aider le général Bol à trouver un plan de bataille permettant de prendre le contrôle de chaque planète alien en le moins de temps possible.

L'efficacité des programmes que vous concevrez sera prise en compte. Vous devrez donc concevoir des structures de données adaptées à vos algorithmes. En particulier accéder rapidement à la liste des planètes voisines à une planète donnée sera important. Il vous sera demandé de donner une réponse pour chacune des situations, que vous ne pourrez calculer dans le temps imparti qu'à la condition que vous ayez programmé des algorithmes efficaces. Il est à noter que tout programme mettant en œuvre les algorithmes et les structures de données préconisés par la suite, auront des temps d'exécutions inférieurs à quelques secondes, pour les jeux de données utilisés dans ce sujet.

2 Génération aléatoire de Galaxies

Une galaxie G représente les forces en présence et correspond donc à la donnée de :

- n le nombre de bases de l'alliance,
- m le nombre de bases alien,
- p le nombre de routes inter-galactiques,
- la liste de $n + m$ couples (n_i, ϱ_i) distincts avec $0 \leq i < n + m$,
- la liste de p triplets $(i, j, \delta_{i,j})$ distincts avec $0 \leq i < n$ et $n \leq j < n + m$.



Dans la suite, toutes les complexités seront exprimées en fonction de n , m et p .
Considérons la suite d'entiers (u_k) définie pour $k \geq 0$ par :

$$u_k = \begin{cases} \text{votre } u_0 \text{ (à reporter sur votre fiche)} & \text{si } k = 0 \\ 15\,091 \times u_{k-1} \pmod{64\,007} & \text{si } k \geq 1 \end{cases}$$

Attention : On choisira u_0 strictement positif et inférieur ou égal à 64 006.

Question 1 Que valent : **a)** u_{10} **b)** u_{100} **c)** u_{1000}

Définition 1 On note $G(n, m, p)$ la galaxie comportant n planètes de l'alliance, m planètes alien et telle que :

- la planète (de l'alliance) i (pour $0 \leq i < n$) dispose de $n_i = u_{2i} \pmod{21}$ vaisseaux au temps $t = 0$ et a un taux de production $\varrho_i = u_{2i+1} \pmod{21}$,
- la planète (alien) i (pour $n \leq i < n + m$) dispose de $n_i = u_{2i} \pmod{7}$ vaisseaux au temps $t = 0$ et a un taux de production $\varrho_i = u_{2i+1} \pmod{7}$,
- la planète (sous le contrôle de l'alliance) $(u_{2(n+m)+3i} \pmod{n})$ est à $(u_{2(n+m)+3i+1} \pmod{50})$ années de la planète (alien) $(n + (u_{2(n+m)+3i+2} \pmod{m}))$ pour $0 \leq i < p$.
Si plusieurs voies inter-galactiques connectent deux planètes, on supprimera évidemment les doublons pour ne retenir que celle dont la durée est la plus courte.

Question 2 Indiquer si la première planète de l'alliance est reliée à la première planète alien (et si oui indiquer le nombre d'années requis pour s'y rendre) dans les galaxies suivantes ?

a) $G(20, 10, 40)$

b) $G(20, 10, 200)$

c) $G(100, 50, 300)$

Un graphe est un couple $G = (S, A)$ d'ensembles satisfaisant $A \subseteq \mathcal{P}_2(S)$ où $\mathcal{P}_2(S)$ est l'ensemble des parties à 2 éléments de S . Les éléments de S sont appelés sommets et ceux de A arêtes. Le degré d'un sommet s est le nombre d'arêtes contenant s .

Un graphe $G = (S, A)$ est dit biparti si S peut être partitionné en deux parties S_1 et S_2 telles que pour tout $\{s, t\}$ appartenant à A : soit $s \in S_1$ et $t \in S_2$, soit $s \in S_2$ et $t \in S_1$. Dans la suite du sujet, on s'intéressera uniquement à des graphes bipartis.

Pour une date de départ t donnée (non nécessairement entière), la planète P_i vaincra la planète P_j si et seulement si le nombre (non nécessairement entier) de vaisseaux partant de P_i au temps t est supérieur ou égal au nombre de défenseurs en P_j au temps $t + \delta_{i,j}$.

Définition 2 On note $G_t(m, n, p)$ le graphe comportant une arête entre P_i et P_j si et seulement si la planète P_i vainc la planète P_j en partant au temps t .

Question 3 Calculer le nombre d'arêtes, ainsi que le degré minimal et maximal des sommets représentant les planètes alien.

a) $G_3(20, 10, 40)$ b) $G_{30}(20, 10, 200)$ c) $G_{50}(100, 50, 300)$

Le graphe G_t varie avec t mais le nombre de dates où ce graphe change est fini. Ces dates sont appelées instants critiques.

Question 4 Calculer les instants critiques et indiquer le premier instant critique strictement positif pour les graphes suivants :

a) $G_3(20, 10, 40)$ b) $G_{30}(20, 10, 200)$ c) $G_{50}(100, 50, 300)$

3 Plan de bataille

On appelle couplage d'un graphe $G = (S, A)$ un sous-ensemble C de A ne contenant aucun couple d'arêtes aboutissant au même sommet de S . La recherche d'un ensemble maximal d'arêtes ayant cette propriété s'appelle un problème de couplage maximal. Il est aisé de voir que l'Alliance remportera une victoire définitive contre les alien en lançant ses vaisseaux au temps t si et seulement si il existe un couplage du graphe G_t de cardinal m .

Pour construire un couplage maximal, on peut générer systématiquement tous les couplages et choisir le meilleur. Cependant cet algorithme "benêt" a pour inconvénient d'avoir une complexité exponentielle en le nombre de routes inter-galactiques, ce qui risque de ne plaire au général en chef des forces de l'alliance. L'objet de cette partie est de concevoir un algorithme plus efficace afin d'éviter son courroux...

Définition 3 On dira qu'une arête $e_1 = \{i_1, j_1\}$ est inférieure à une arête $e_2 = \{i_2, j_2\}$ (avec i_1, j_1 des sommets correspondant à une planète de l'alliance et i_2, j_2 des sommets correspondant à une planète alien) si et seulement si (i_1, j_1) est inférieur à (i_2, j_2) pour l'ordre lexicographique.

Pour un graphe G donné, on note E l'ensemble trié des arêtes de G et C l'ensemble trié d'arêtes extrait de E en en sélectionnant une sur 10.

Question 5 Donner la première arête de C et indiquer si C est un couplage ou non pour les ensembles suivants :

a) $C_3(20, 10, 40)$ b) $C_{30}(20, 10, 200)$ c) $C_{50}(100, 50, 300)$

Une suite de sommets distincts (x_0, x_1, \dots, x_p) de G est appelée chaîne de longueur p si $\{x_0, x_1\}, \{x_1, x_2\}, \dots, \{x_{p-1}, x_p\}$, sont des arêtes de G .

Soit C un couplage de G . Un sommet est dit saturé si c'est l'extrémité d'une arête de C (on dira sinon qu'il est insaturé). Une chaîne dans laquelle une arête sur deux est dans C et les autres dans $A \setminus C$ est dite alternée et une chaîne alternée reliant deux sommets insaturés par C est dite améliorante. Notons au passage qu'une chaîne améliorante a toujours une longueur impaire.

Étant donné un couplage C et une chaîne améliorante C_A , il est toujours possible de construire un couplage de cardinalité supérieure en retirant dans C les sommets présents dans C_A et en insérant dans C les arêtes de C_A qui n'appartenaient pas à C . Ainsi, le nouveau couplage est composé des arêtes présentes dans C ou dans C_A mais pas dans les deux simultanément

Question à développer pendant l'oral : Montrer qu'un couplage C est maximal s'il n'accepte aucune chaîne améliorante.

On peut alors en déduire un algorithme pour construire un couplage maximal en supposant que l'on sache construire une chaîne améliorante pour un couplage donné si elle existe. Reste donc à montrer comment construire une chaîne améliorante pour un couplage C donné.

Pour trouver une chaîne améliorante, il suffit de parcourir récursivement le graphe en partant des sommets de S_2 non saturés et en se déplaçant alternativement sur des arêtes n'appartenant pas au couplage et sur des arêtes y appartenant. À partir du moment où on arrive sur un sommet de S_1 non saturé, on a obtenu une chaîne améliorante. Lors de l'exploration, les sommets seront toujours explorés en allant de celui de plus petit indice à celui ayant le plus grand indice. Pour parcourir le graphe, il sera certainement utile d'avoir un tableau indiquant si un sommet a déjà été visité et s'il est saturé ou non.

Question 6 Si l'ensemble C précédemment défini est un couplage, construire une chaîne améliorante et donner le plus petit indice de cette chaîne pour les graphes suivants :

a) $G_3(20, 10, 40)$ **b)** $G_{30}(20, 10, 200)$ **c)** $G_{50}(100, 50, 300)$

Question 7 Fusionner C et la chaîne améliorante précédente (si elle existe) et donner le plus petit indice des sommets couverts par C pour les graphes suivants :

a) $G_3(20, 10, 40)$ **b)** $G_{30}(20, 10, 200)$ **c)** $G_{50}(100, 50, 300)$

Question 8 Calculer un couplage maximal des graphes suivants et indiquer leur taille :

a) $G_3(20, 10, 40)$ **b)** $G_{30}(20, 10, 200)$ **c)** $G_{50}(100, 50, 300)$

4 Calcul du jour J

Question 9 Calculer le premier instant t permettant de prendre le contrôle des planètes aliens pour les galaxies suivantes :

a) $G(20, 10, 40)$ **b)** $G(20, 10, 200)$ **c)** $G(100, 50, 300)$

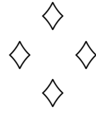
On enlève maintenant la contrainte que tous les attaquants doivent partir de leur planète à la même date. En revanche, une fois qu'une planète a envoyé des attaquants, elle ne peut plus participer à une autre bataille.

Question 10 *Calculer le premier instant t permettant de prendre le contrôle des planètes aliens pour les galaxies suivantes :*

a) $G(20, 10, 40)$

b) $G(20, 10, 200)$

c) $G(100, 50, 300)$



La Guerre des Étoiles

Nom, prénom, u₀:

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)

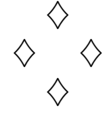
c)

Question 10

a)

b)

c)



Fiche d'évaluation: La Guerre des Étoiles

Nom, prénom, u₀:

Question 1 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

Question 2 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

Question 3 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

Question 4 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

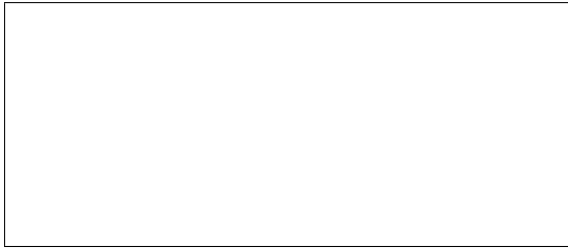
Question 5 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

Question 6 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|4
- b) 0|-----|-----|-----|4
- c) 0|-----|-----|-----|4

Question 7 0-1-2-3-4-5-6-7-8

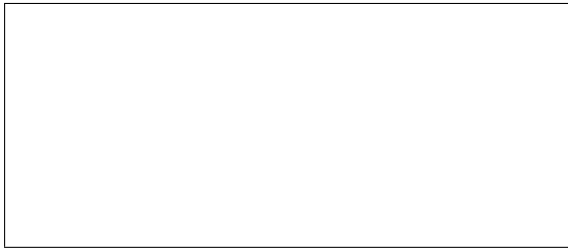


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 8 0-1-2-3-4-5-6-7-8

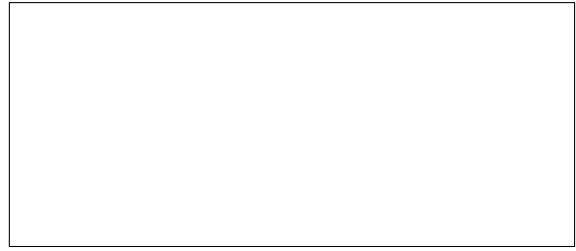


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 9 0-1-2-3-4-5-6-7-8



a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 10 0-1-2-3-4-5-6-7-8



a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4



Dératisation à Manhattan

Épreuve pratique d'algorithmique et de programmation

Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin/juillet 2007

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*. 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 Introduction

Baaaam! Encore une explosion de gaz dans les sous-sols de Manhattan. En effet, ces dernières années, d'impressionnantes colonies de rats ont élu domicile dans les égouts et le service de dératisation fait son possible pour éliminer ce fléau.

Comme vous le savez, Manhattan est organisé de façon très régulière avec des rues et des avenues formant une grille rectangulaire. Les égouts suivent le même arrangement et les rats établissent toujours leurs colonies au niveau des intersections de rues. La seule méthode efficace pour s'en débarrasser semble être l'utilisation de bombes de gaz semblables à celle qui vient d'exploder. Cependant, ce gaz n'est pas nocif que pour les rats et il convient d'évacuer les immeubles dans la zone d'action de la bombe avant son explosion. Le placement de ces bombes doit donc être choisi avec précaution. Une des spécificités de ces bombes est que lors de leur explosion, le gaz se diffuse de façon rectangulaire le long des égouts. La Figure 1 représente la zone couverte par l'explosion d'une bombe de rayon 1.

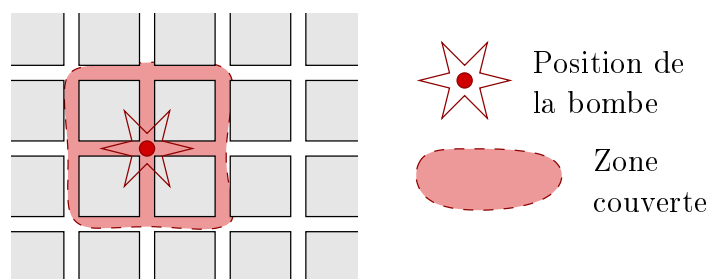


FIG. 1 – Zone couverte par l'explosion d'une bombe de rayon 1.

Notre objectif est de réfléchir à la conception d'algorithmes permettant soit de supprimer le plus grand nombre de rats à l'aide d'un ensemble de bombes donné, soit de trouver le plus petit nombre de bombes (de même rayon) nécessaires à l'extinction de toutes les colonies.

L'efficacité des programmes que vous concevrez sera prise en compte. Il vous sera demandé de donner une réponse pour chacune des situations, que vous ne pourrez calculer dans le temps imparti qu'à la condition que vous ayez programmé des algorithmes efficaces. Il est à noter que moins d'une minute de calcul est nécessaire pour chacune des questions de ce sujet.

2 Génération aléatoire de colonies

La situation de la ville de Manhattan peut être modélisée par les données suivantes :

- n le nombre de colonies,
- m le nombre de bombes,
- la liste de n triplets (x_i, y_i, t_i) avec $1 \leq i \leq n$ où (x_i, y_i) représente la coordonnée de la $i^{\text{ème}}$ colonie et t_i la taille de cette colonie,
- le rayon d des m bombes.

Chacune des valeurs précédentes est à valeur dans \mathbb{N}^* .

L'objectif est de placer les m bombes dans le plan afin de supprimer le plus grand nombre de rats possible. Un placement P des bombes est représenté par $(X_1, Y_1), \dots, (X_m, Y_m)$. Une colonie est couverte par le placement P s'il existe une bombe j à distance (pour la norme infinie) inférieure à d . Autrement dit, la colonie i est couverte par le placement P si et seulement s'il existe une bombe j telle que $\max(|x_i - X_j|, |y_i - Y_j|) \leq d$.

Le poids d'un placement est la somme des t_i des colonies couvertes (attention, une colonie couverte par plusieurs bombes ne compte qu'une seule fois dans le poids d'un placement) et c'est donc ce que l'on va chercher à maximiser par la suite.

Considérons la suite d'entiers (u_k) définie pour $k \geq 0$ par :

$$u_k = \begin{cases} \text{votre } u_0 \text{ (à reporter sur votre fiche)} & \text{si } k = 0 \\ 15\,091 \times u_{k-1} \pmod{64\,007} & \text{si } k \geq 1 \end{cases}$$

Attention : On choisira u_0 strictement positif et inférieur ou égal à 64 006.

Question 1 Que valent : **a)** u_{10} **b)** u_{100} **c)** u_{1000} .

Définition 1 On note $C(n, m, d)$ l'instance comportant m bombes de rayon d et n colonies telles que la colonie i (pour $0 \leq i < n$) est située en $(x_i, y_i) = ((u_{3i} \pmod{10000}) + 1, (u_{3i+1} \pmod{10000}) + 1)$ et est de taille $t_i = (u_{3i+2} \pmod{40}) + 1$.

S'il existe $i < j$ tel que $(x_i, y_i) = (x_j, y_j)$ on fusionnera la colonie j avec la colonie i en ajoutant t_j à t_i et on supprimera la colonie j de la liste. Il y aura alors strictement moins de n colonies.

Question 2 Donner la taille de la colonie ayant la plus petite coordonnée selon x (et en cas d'ex-aequo celle ayant la plus petite coordonnée selon y).

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

3 Le cas mono-dimensionnel

Dans cette partie, on s'intéresse au cas mono-dimensionnel, c'est-à-dire au cas où on considère que $y_i = 1$ pour tout $0 \leq i < n$. On ne considèrera que des placements $[x - d, x + d]$ dits "canoniques", c'est-à-dire tels que $x - d$ coïncide avec une colonie et x est le plus petit possible.

Question 3 Écrire une fonction retournant le placement optimal d'une bombe et indiquer le nombre de rats tués pour chacune des configurations suivantes :

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 4 Écrire une fonction retournant le placement optimal de deux bombes et indiquer le nombre de rats tués pour chacune des configurations suivantes :

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 5 Écrire une fonction calculant le nombre minimal de bombes pour éradiquer toutes les colonies et indiquer ce nombre pour chacune des configurations suivantes :

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

On peut remarquer un certain nombre de choses. D'une part le nombre de positions canoniques pour une bombe donnée est borné par n . D'autre part, le placement optimal de m bombes peut s'obtenir en posant une bombe à un endroit donné et en utilisant le placement optimal de $m - 1$ bombes pour les colonies non couvertes et situées à droite de la première bombe. Si on note $A[s, k]$ le nombre optimal de rats tués en plaçant k bombes à droite de la s -ième colonie, on peut donc écrire une relation de récurrence sur A et le stockage de A ne nécessite qu'un espace mémoire raisonnable.

Question 6 *En déduire un algorithme retournant un placement optimal de m bombes et indiquer le nombre de rats tués pour chacune des configurations suivantes :*

- a)** $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

4 Le cas bi-dimensionnel

On s'intéresse dans cette partie au cas général où les deux coordonnées sont considérées. On peut alors faire le constat suivant :

- comme l'illustre la Figure 2, on peut se restreindre aux positions de bombes telles que le bord supérieur et le bord gauche de la zone de couverture sont tous deux en contact avec une colonie (pas nécessairement la même).

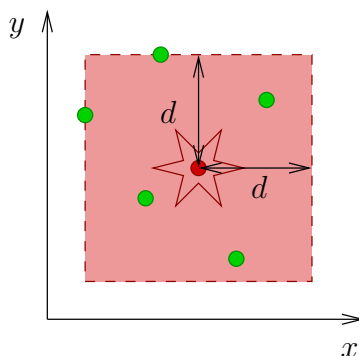


FIG. 2 – Positionnement canonique d'une bombe : le bord supérieur et le bord gauche de la zone de couverture sont tous deux en contact avec une colonie.

Le nombre de positionnement est donc toujours polynomial mais bien plus gros que précédemment. On propose de regarder l'algorithme glouton consistant à placer successivement les bombes à l'endroit permettant d'éliminer le plus grand nombre de rats possible.

On propose donc l'utilisation de la structure suivante pour essayer de réduire au mieux la complexité. On note $R = \{1, \dots, n\}$ l'ensemble des colonies B l'ensemble de positionnements canoniques de bombes possibles. On dira que $r \in R$ et $b \in B$ sont reliés si le positionnement b couvre la colonie r . On note E l'ensemble de ces relations. Pour mettre en oeuvre efficacement l'algorithme glouton on aura besoin de savoir rapidement le nombre de rats tués par une bombe donnée. On aura également besoin de pouvoir mettre à jour rapidement R , B et E lorsque l'on pose une bombe donnée.

Question 7 *Écrire une fonction construisant la structure en question et indiquer $|B|$, le nombre de positions possibles pour les configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 8 *Écrire une fonction retournant le placement optimal d'une bombe et indiquer le nombre de rats tués pour chacune des configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 9 *Écrire une fonction mettant la structure à jour lorsque l'on pose une bombe à un endroit donné et écrire l'algorithme glouton plaçant m bombes. Indiquer le nombre de rats tués pour chacune des configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 10 *Écrire une fonction calculant gloutonnement le nombre de bombes pour éradiquer toutes les colonies et indiquer ce nombre pour chacune des configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

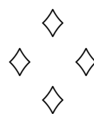
Suite aux temps de calcul prohibitifs et aux résultats décevants de cet algorithme, on propose une autre approche. L'idée consiste à découper l'espace en bandes horizontales de largeur $2d$ et à utiliser les solutions pour le cas mono-dimensionnel. Attention, pour créer ces bandes horizontales, on prendra bien soin d'utiliser le même type de techniques que dans le cas mono-dimensionnel et de toujours traiter les coordonnées par ordre croissant.

Question 11 *Pour couvrir toutes les colonies, on propose d'appliquer la solution optimale mono-dimensionnelle dans chaque bande. Indiquer le nombre de bombes nécessaires à l'aide de cette méthode pour chacune des configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$

Question 12 *On propose un autre algorithme pour placer m bombes. Comme précédemment, on coupe en bandes horizontales de largeur $2d$, puis on sélectionne la bande où il est le plus intéressant de placer une bombe. On met à jour et on recommence. Indiquer le nombre de rats tués à l'aide de cette méthode pour chacune des configurations suivantes :*

a) $C(10, 2, 500)$ **b)** $C(50, 10, 500)$ **c)** $C(1000, 20, 500)$



Dératisation à Manhattan

Nom, prénom, u₀:

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)

Question 11

a)

b)

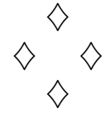
c)

Question 12

a)

b)

c)



Fiche d'évaluation: Dératisation à Manhattan

Nom, prénom, u₀:

Question 1 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 2 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 3 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 4 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

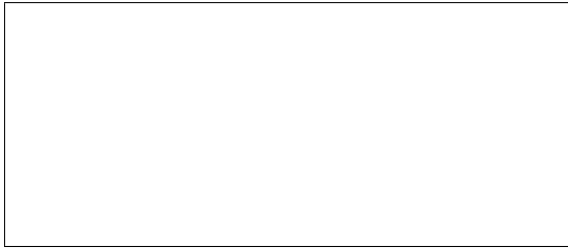
Question 5 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 6 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 7 0-1-2-3-4-5-6-7-8

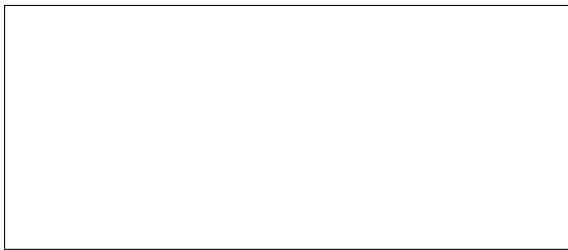


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 8 0-1-2-3-4-5-6-7-8

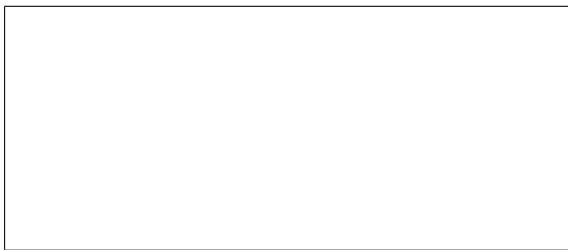


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 9 0-1-2-3-4-5-6-7-8

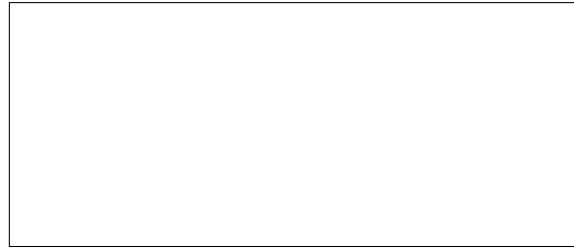


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 10 0-1-2-3-4-5-6-7-8

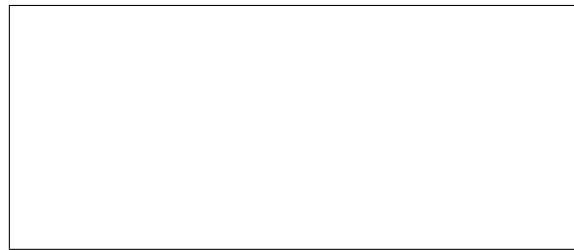


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 11 0-1-2-3-4-5-6-7-8

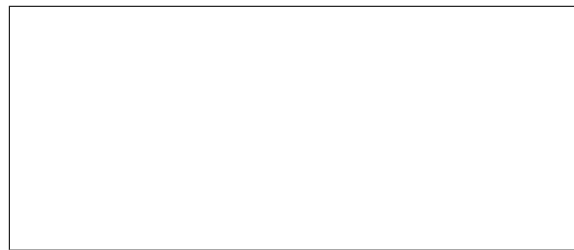


a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 12 0-1-2-3-4-5-6-7-8



a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4



Surface aléatoire

Épreuve pratique d'algorithmique et de programmation

Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juin/juillet 2007

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.*

Nous étudions dans cette épreuve une surface représentée par un tableau à deux dimensions d'entiers. Cette surface est définie de façon pseudo-aléatoire, de façon telle que la variation de hauteur entre deux points voisins reste modérée.

1 Préliminaire

Soit la suite $(u_n)_{n \geq 0}$ définie de la façon suivante :

- u_0 est donné sur votre table (à reporter sur votre fiche).
- $u_{n+1} = (1 + 16423 \times u_n) \text{ modulo } 62951$.

Question 1 Quelle est la valeur de **a)** u_5 , **b)** u_{1000} , **c)** u_{40000} ?

On pose ensuite $v_n = (u_n \text{ modulo } 31) - 15$.

Question 2 Quelle est la valeur maximale de $|\sum_{k=0}^n v_k|$ pour **a)** $n \leq 5$, **b)** $n \leq 1000$, **c)** $n \leq 40000$?

2 Définitions et notations

Dans la suite de l'énoncé, les définitions et notations seront utilisées :

- n étant un entier fixé, on définit $\mathbb{N}_n = \{0, \dots, n-1\}$ et $C_n = \mathbb{N}_n \times \mathbb{N}_n$;
- deux éléments (x, y) et (x', y') de C_n sont dit adjacents si $|x - x'| + |y - y'| = 1$; plus généralement, la distance entre (x, y) et (x', y') sera définie comme $D_{(x,y),(x',y')} = |x - x'| + |y - y'|$;
- si x est un réel, $[x]$ désigne l'entier le plus proche de x (l'arrondi se fait par défaut si la partie fractionnaire x est strictement inférieur à 0,5, et par excès sinon) ;
- si E est un ensemble, $|E|$ désigne le cardinal de E .

3 Fonction

On définit une fonction h de C_n dans \mathbb{Z} à partir de la suite v par les relations suivantes :

- $h(0, 0) = 1000$.
- pour $1 \leq i \leq n-1$,

$$\begin{aligned} h(i, 1) &= h(i-1, 1) + v_{i^2} \\ h(1, i) &= h(1, i-1) + v_{i^2+1} \end{aligned}$$

- pour $1 \leq i \leq n$ et $1 \leq j \leq i$,

$$h(i, j) = \left[\frac{4 \cdot (h(i-1, j) + h(i, j-1)) - 3 \cdot h(i-1, j-1)}{5} \right] + v_{i^2+2j}$$

et, si $j \neq i$,

$$h(j, i) = \left[\frac{4 \cdot (h(j-1, i) + h(j, i-1)) - 3 \cdot h(j-1, i-1)}{5} \right] + v_{i^2+2j+1}$$

On remarquera que, dès lors que i et j sont inférieurs à n , $h(i, j)$ est indépendant de n .

Question 3 Avec $n = 200$, donnez les valeurs de : **a)** $h(2, 1)$, **b)** $h(19, 13)$, **c)** $h(199, 199)$? Quelle est l'amplitude (la différence entre le maximum et le minimum) de h sur C_n pour **d)** $n = 5$; **e)** $n = 50$; **f)** $n = 200$? Quel est l'écart maximum de h entre deux points adjacents de C_n pour **g)** $n = 5$; **h)** $n = 50$; **i)** $n = 200$?

4 Éclairage

On suppose notre surface éclairée par un source à l'infini émettant des rayons lumineux parallèles. Par définition, un point (i, j) sera éclairé si :

$$\forall 0 \leq k < \min\{i, j\}, h(i, j) > h(i - k, j - k) - 3k.$$

Question 4 Donner le nombre de points de C_n éclairés pour **a)** $n = 5$; **b)** $n = 50$; **c)** $n = 200$.

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

5 Chemins

On définit un chemin dans C_n de longueur l comme une fonction c de $\{1, \dots, l\}$ dans C_n telle que pour tout i dans $\{1, \dots, l - 1\}$, $c(i)$ et $c(i + 1)$ sont adjacents. L'origine du chemin est $c(1)$, sa destination $c(l)$.

Dans ce qui suit, si c est un chemin dans C_n , x^c et y^c désignent les deux composantes de c .

Pour assurer l'unicité des réponses dans les questions à venir, on utilise un ordre (lexicographique) \preceq sur les chemins : $c \preceq c'$ si le plus petit indice i_0 tel que $c(i_0) \neq c'(i_0)$ vérifie soit $x^c(i_0) < x^{c'}(i_0)$, soit $x^c(i_0) = y^{c'}(i_0)$ et $y^c(i_0) < y^{c'}(i_0)$.

Nous considérons l'ensemble P_n des chemins allant de $(0, 0)$ à $(n - 1, n - 1)$, de sorte que pour tout i , $x^c(i + 1) - x^c(i) = 1$ ou $y^c(i + 1) - y^c(i) = 1$. Ainsi les chemins de P_n sont tous de longueurs $2n - 1$.

Pour chaque chemin c de longueur, on définit une fonction de coût f :

$$f(c) = \sum_{i=2}^l |h(c(i)) - h(c(i - 1))|$$

Notre but est d'abord de construire un chemin de P_n ayant le plus grand coût possible, en partant de $(0, 0)$, en supposant connue les valeurs de h uniquement de façon locale. Ainsi, k étant un entier strictement positif fixé, on suppose qu'à partir d'un point (x, y) , on ne peut accéder aux valeurs de h que pour les points (x', y') situés à une distance inférieure ou égale à k de (x, y) . On choisit alors le point adjacent de (x, y) commençant un chemin de longueur k de coût maximal pouvant s'intégrer dans un chemin de P_n .

Lorsqu'il y a plusieurs possibilités équivalentes, on privilégiera le chemin maximal selon l'ordre \preceq .

Question 5 Donner, pour les valeurs de n, k suivantes, le coût du chemin total construit
a) $n = 5$ et $k = 2$; **b)** $n = 50$ et $k = 5$; **c)** $n = 200$ et $k = 50$.

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

On cherche à présent un chemin de coût minimal allant de $(0, 0)$ à $(n - 1, n - 1)$, sans la contrainte que ce chemin doit appartenir à P_n (ainsi sa longueur est quelconque). La méthode « classique » pour construire un tel chemin est de calculer pour chaque point le coût du chemin de coût minimal y arrivant. Pour cela, on maintient deux ensembles de sommets E et F , et un majorant $m(x, y)$ du coût du chemin minimal permettant d'aller de $(0, 0)$ à (x, y) .

- Initialement F est l'ensemble des points, et E est vide. On sait d'autre part qu'un chemin de coût 0 permet d'aller de $(0, 0)$ à $(0, 0)$.
- À chaque étape, on choisit le point (x, y) de F ayant le plus petit $m(x, y)$, on retire ce point de F et on l'ajoute dans E . De plus, on mets à jour les valeurs de $m(x', y')$ pour les points (x', y') dans F adjacents à (x, y) , en prenant en compte les chemins passant par (x, y) .
- Notre algorithme s'arrête lorsque $(n - 1, n - 1)$ n'est plus dans F .

Ainsi, à tout instant, $m(x, y)$ représente le coût du chemin de coût minimal (lorsqu'il existe) de $(0, 0)$ à (x, y) , passant uniquement par des points de E (sauf pour sa destination).

Pour améliorer l'algorithme, on peut prendre en compte le fait que le coût pour passer d'un point à un point adjacent ne prend qu'un nombre limité de valeurs, ce qui limite le nombre de valeurs possibles à un instant donné pour $m(x, y)$ avec (x, y) dans F .

Question 6 Donner le coût du chemin de coût minimal d'origine $(0, 0)$ et de destination $(n - 1, n - 1)$ pour **a)** $n = 5$, **b)** $n = 50$ et **c)** $n = 200$.

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

6 Minima

Un couple (i, j) avec $1 \leq i \leq n$ et $1 \leq j \leq n$ est un minimum local pour h si $h(i, j)$ est inférieur ou égal à tous les $h(i', j')$ tels que (i', j') soient adjacents à (i, j) . On note M l'ensemble de ces minima.

On considère les chemins « descendant » (c'est-à-dire tels que $h \circ c$ est décroissante au sens large) dont la destination est un minimum local. On note $D_{(i,j)}$ l'ensemble de ces chemins d'origine (i, j) , et $M_{(i,j)}$ l'ensemble des destinations des chemins dans $D_{(i,j)}$.

Question 7 Combien vaut $\max_{(i,j) \in C_n} |M_{(i,j)}|$, c'est-à-dire le nombre maximum de minima accessibles depuis par un chemin descendant depuis un point donné, pour **a)** $n = 5$, **b)** $n = 50$, et **c)** $n = 200$? Combien de points P permettent d'atteindre un tel nombre de minima pour **d)** $n = 5$, **e)** $n = 50$, et **f)** $n = 200$?

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

On définit à présent une structure de graphe non orienté pondéré sur M : deux éléments distincts (x, y) et (x', y') de M sont reliés par une arête de poids a si :

- il existe un chemin c d'origine (x, y) , de destination (x', y') , croissant jusqu'à un certain indice i_c puis décroissant au-delà ;
- a est le minimum des $h(c(i_c))$, où c est l'ensemble des chemins vérifiant les propriétés ci-dessus.

Ainsi, une arête de poids a lie deux minima « voisins » par un col de hauteur a . On note A l'ensemble des arêtes du graphe, la $w : A \rightarrow \mathbb{Z}$ la fonction de pondération. Aucune arête ne relie un sommet à lui-même.

Question 8 *Quel est le cardinal de A pour a) $n = 5$? b) $n = 50$? c) $n = 200$?*

Etant donné un entier k strictement positif, on considère \mathcal{Z}_k l'ensemble des sous-ensembles Z non vide de M vérifiant la propriété suivante : il existe $(x, y) \in Z$ vérifiant $h(x, y) = \min h(Z)$, et Z est l'ensemble des éléments de M accessibles depuis (x, y) par un chemin c tel que le maximum de $h \circ c$ est inférieur ou égal à $h(x, y) + k$. Noter que \mathcal{Z}_k est toujours non vide.

Question 9 *Avec $k = 30$, quel est le cardinal de \mathcal{Z}_k pour a) $n = 5$, b) $n = 50$ et c) $n = 200$?*

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

Question 10 *Quelle est la plus petite valeur de k tel que \mathcal{Z}_k est un singleton pour a) $n = 5$, b) $n = 50$ et c) $n = 200$?*

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



Surface aléatoire

Nom, prénom, u_0 :

Question 1

a)

b)

c)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

d)

e)

f)

g)

h)

i)

Question 4

a)

b)

c)

Question 5

a)

b)

c)

Question 6

a)

b)

c)

Question 7

a)

b)

c)

d)

e)

f)

Question 8

a)

b)

c)

Question 9

a)

b)

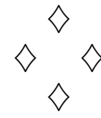
c)

Question 10

a)

b)

c)



Fiche d'évaluation: Surface aléatoire

Nom, prénom, u₀:

Question 1 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4

Question 2 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4

Question 3 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4
- d) 0|-----|4
- e) 0|-----|4
- f) 0|-----|4

- g) 0|-----|4
- h) 0|-----|4
- i) 0|-----|4

Question 4 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4

Question 5 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4

Question 6 0-1-2-3-4-5-6-7-8

- a) 0|-----|4
- b) 0|-----|4
- c) 0|-----|4

Question 7 0-1-2-3-4-5-6-7-8

- a) 0 |-----| 4
- b) 0 |-----| 4
- c) 0 |-----| 4
- d) 0 |-----| 4
- e) 0 |-----| 4
- f) 0 |-----| 4

Question 8 0-1-2-3-4-5-6-7-8

- a) 0 |-----| 4
- b) 0 |-----| 4
- c) 0 |-----| 4

Question 9 0-1-2-3-4-5-6-7-8

- a) 0 |-----| 4
- b) 0 |-----| 4
- c) 0 |-----| 4

Question 10 0-1-2-3-4-5-6-7-8

- a) 0 |-----| 4
- b) 0 |-----| 4
- c) 0 |-----| 4



Le Réchauffement Climatique

Épreuve pratique d'algorithmique et de programmation

Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juillet 2007

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 Introduction

Comme chacun sait, le climat se réchauffe, ce qui a des conséquences sur la montée des eaux des océans. Les experts scientifiques mandatés par l'ONU aimeraient étudier plus précisément quelles villes seraient impactées dans les divers scénarios qu'ils envisagent. Pour ce faire, ils disposent de la liste des villes, et pour chacune, de l'intervalle fermé d'altitude sur lequel elle s'étend.

Ils aimeraient en particulier pouvoir déterminer, pour une prévision de montée des eaux à une altitude donnée, les villes qui seraient partiellement inondées, c'est-à-dire où le niveau d'eau serait compris dans l'intervalle d'altitude.

Nous allons donc étudier des structures de données permettant de répondre à ces questions de manière efficace, à savoir les listes à enjambements et des extensions pour la recherche d'intervalles. Ces structures permettent de faire des recherches rapides dans des ensembles d'objets triés, et elles peuvent être mises à jour facilement lors d'insertions de nouveaux éléments.

Les réponses à certaines questions ne peuvent être calculées que si vos algorithmes sont efficaces. Il est également important de suivre la méthode recommandée pour l'implémentation, car elle sera fondamentale dans la suite du sujet.

2 Recherche simple

Nous allons tout d'abord programmer la liste à enjambements classique, qui permet de faire des recherches rapides dans un ensemble d'éléments triés. L'idée de départ est simple. Considérons une liste simplement chaînée de n éléments triés en ordre croissant. Dans une telle liste, rechercher un élément nécessitera au pire n comparaisons. Insérer un élément prend le temps d'une recherche, plus un temps constant de mise à jour de la liste.

Considérons maintenant l'idée suivante : afin d'accélérer la recherche, nous ajoutons aux pointeurs classiques de la liste, une série de pointeurs qui permettent de faire des sauts plus longs dans la liste. Pour commencer, les éléments de rang pair dans la liste seront connectés eux aussi sous forme de sous-liste chaînée. De cette façon, la recherche peut maintenant commencer par une première étape de recherche dans cette sous-liste, qui ne compte que $n/2$ éléments, puis descendre dans la liste initiale où il n'y a plus qu'une seule comparaison à effectuer.

De manière récursive, cette sous-liste peut elle-même avoir une sous-liste, etc. On dit que la liste principale est de niveau 0, sa première sous-liste de niveau 1, etc.

Pour $n = 2^k$ (pour k entier), on obtient donc $k + 1$ listes de niveaux $0 \dots k$. On définit le niveau d'un élément comme étant celui de la liste de plus haut niveau qui le contient. On peut aussi remarquer que la liste de niveau k contient exactement les éléments de niveau au moins k .

La figure 1 donne un exemple d'une telle structure de donnée.

Question à développer pendant l'oral : Quelle est la complexité en temps d'une recherche ?

Le problème de cette structure est la difficulté de sa mise à jour lors d'insertions de nouveaux éléments. Pour améliorer cet aspect, nous allons conserver la propriété importante

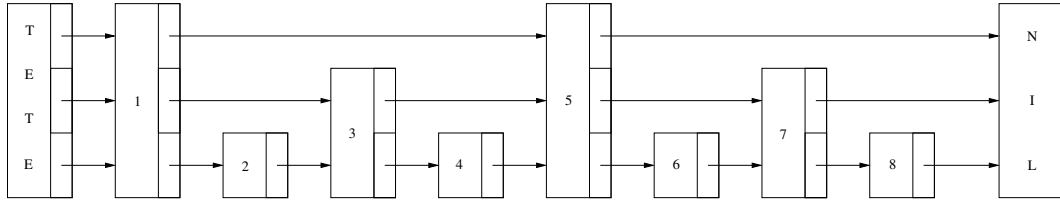


FIG. 1 – Exemple de structure de recherche fixe contenant les éléments $\{1..8\}$ de niveaux respectifs $\{2, 0, 1, 0, 2, 0, 1, 0\}$. La sous-liste de niveau 1 contient les éléments $\{2, 4, 6, 8\}$.

de la structure, à savoir le nombre exponentiellement décroissant d'éléments par niveau, mais au lieu d'avoir des sous-listes régulièrement espacées difficiles à maintenir, nous allons randomiser la structure, en utilisant une distribution aléatoire des éléments à chaque niveau. Ainsi, lors de l'insertion d'un élément, son niveau sera choisi aléatoirement selon une distribution adaptée, et il conservera toujours ce niveau quels que soient les éléments insérés plus tard.

Pour ce faire, nous choisissons donc un générateur aléatoire de niveaux qui fournisse une distribution des tailles de chaque niveau qui décroisse exponentiellement. En pratique, on bornera le niveau maximum généré par la valeur 32 afin de simplifier la programmation.

La figure 2 donne un exemple d'une telle structure de donnée.

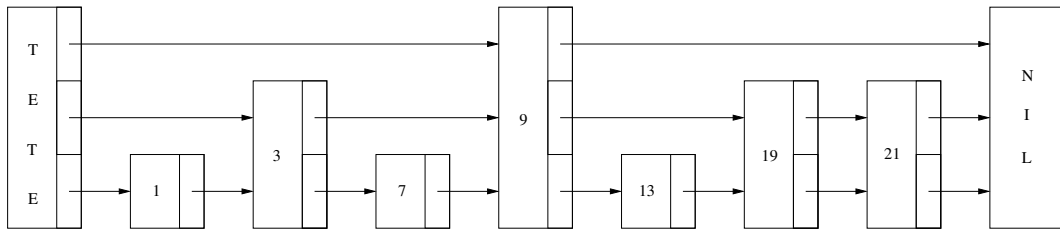


FIG. 2 – Exemple de liste à enjambements contenant les éléments $\{1, 3, 7, 9, 13, 19, 21\}$ de niveaux respectifs $\{0, 1, 0, 2, 0, 1, 1\}$. La sous-liste de niveau 1 contient les éléments $\{3, 9, 19, 21\}$.

3 Suite pseudo-aléatoire de niveaux

Considérons les suites d'entiers (u_n) , (v_n) et (w_n) définies pour $n \geq 0$ par :

$$u_n = \begin{cases} \text{votre } u_0 \text{ (à reporter sur votre fiche)} & \text{si } n = 0 \\ 15\,091 \times u_{n-1} \pmod{64\,007} & \text{si } n \geq 1 \end{cases}$$

$$v_n = \begin{cases} u_0 & \text{si } n = 0 \\ 1\,129 \times v_{n-1} \pmod{15\,193} & \text{si } n \geq 1 \end{cases}$$

$$w_n = u_n + 64\,007 \times v_n$$

Attention : On choisira u_0 strictement positif et inférieur ou égal à 15 192.

Question 1 Que valent : **a)** w_{10} **b)** $w_{1\,000}$ **c)** $w_{100\,000}$

Étant donné un réel $p \in [0; 1]$, on considère la suite (p_n) à valeurs dans $\{0, 1\}$ définie par :

$$p_n = \begin{cases} 0 & \text{si } w_n < p \times (64007 \times 15193) \\ 1 & \text{sinon} \end{cases}$$

Dans la suite, nous considérerons $p = \frac{1}{2}$.

Nous définissons maintenant la suite (l_n) . Considérons les valeurs $\{p_{32n+i}\}$, pour i dans $\{0, \dots, 31\}$. l_n est définie comme étant la valeur de i la plus petite pour laquelle la valeur de $\{p_{32n+i}\}$ vaut 1, ou 32 si aucune de ces valeurs ne vaut 1.

La suite (l_n) va servir de générateur pseudo-aléatoire de niveaux pour l'insertion de nouveaux éléments dans la liste à enjambements.

Question 2 Que valent : **a)** l_{10} **b)** $l_{1\,000}$ **c)** $l_{100\,000}$

Question 3 Distribution des valeurs de (l_n) : parmi $\{l_0, \dots, l_{10\,000}\}$, combien de fois apparaissent les valeurs : **a)** 1 **b)** 3 **c)** 10

Question à développer pendant l'oral : Quelle est la distribution des valeurs de (l_n) et qu'est-ce que cela suggère sur la complexité en moyenne de l'insertion d'un élément dans la liste à enjambements ?

Question à développer pendant l'oral : Dérandomisation : il est parfois intéressant d'introduire du déterminisme dans la structure de données, sans toutefois perdre les propriétés de distribution des niveaux. Proposez un moyen simple qui permette de garantir que le niveau d'un élément ne dépende que de sa valeur (on ne demande pas de le programmer).

4 Structure de recherche classique

Nous demandons maintenant de programmer la structure de données décrite, ainsi que la procédure de recherche d'un élément et celle d'insertion. La procédure de recherche retournera les noeuds de chaque niveau dont la valeur précède la valeur recherchée, de sorte à pouvoir être utilisée par la procédure d'insertion. À chaque insertion d'un élément, une valeur de la suite (l_n) sera utilisée comme niveau d'insertion (w_i aura donc le niveau l_i , dans les questions de cette section 4). Si un élément à insérer existe déjà dans la structure, il n'y sera stocké qu'une seule fois.

Pour tester, nous allons tout d'abord utiliser la structure pour y insérer successivement les valeurs $\{w_0, \dots, w_{10}\}$. On suggère fortement de vérifier le contenu de la structure sur ce petit exemple (en affichant par exemple le contenu de chaque sous-liste après chaque insertion).

Question 4 Dans la liste à enjambements ainsi obtenue, quel est le rang des éléments suivants (le premier élément étant de rang 0), dans la liste principale de niveau 0 ? **a)** w_0 **b)** w_1 **c)** w_{10}

Utilisez maintenant la structure pour y insérer successivement les valeurs $\{w_0, \dots, w_{100\,000}\}$.

Question 5 Dans la liste à enjambements ainsi obtenue, quel est le rang des éléments suivants, dans la liste principale de niveau 0? **a)** w_{10} **b)** $w_{1\,000}$ **c)** $w_{100\,000}$

Question 6 Dans la liste à enjambements ainsi obtenue, quel est le rang des éléments suivants dans la sous-liste de leur propre niveau? **a)** w_{10} **b)** $w_{1\,000}$ **c)** $w_{100\,000}$

5 Extension à la recherche d'intervalles contenant une valeur

Considérons maintenant le problème réel, où la structure de données ne stocke pas simplement des valeurs, mais des intervalles de valeurs. Ces intervalles peuvent bien sûr se chevaucher, ils ne sont donc pas totalement ordonnés, même si l'ensemble de leurs bornes l'est. La structure de données doit maintenant pouvoir répondre rapidement l'ensemble des intervalles qui contiennent une valeur donnée.

Nous proposons maintenant de programmer une telle structure de données, avec notamment un algorithme d'insertion de nouvel intervalle en s'inspirant de la liste à enjambements. Pour cela, nous allons commencer par utiliser la liste à enjambements pour stocker les 2 bornes de chaque intervalle. Pour des raisons d'efficacité, il est impératif qu'un intervalle ne soit pas systématiquement représenté à un coût proportionnel au nombre d'éléments situés entre ses deux bornes. On utilisera donc la capacité d'"enjambement" de la structure dans ce but.

La figure 3 donne un schéma de la représentation attendue. En particulier, la recherche d'une valeur dans la structure, qui utilise le même algorithme que précédemment, fournit la liste des intervalles recherchés. Un intervalle est donc stocké tout le long d'une chaîne d'éléments, dont les niveaux montent puis redescendent, et ce, dans les niveaux les plus élevés possibles afin de minimiser le stockage. Dans l'exemple, l'intervalle $[1; 9]$ est ainsi stocké dans le noeud 1 au niveau 1, puis dans le noeud 3 au niveau 2, puis 7 au niveau 2, et finalement 9 au niveau 1. On a donc évité de le stocker dans les noeuds 5 et 6, ce qui est le but recherché. Lors de l'insertion d'un nouvel intervalle, on fera bien attention à mettre à jour les noeuds référençant les intervalles existants.

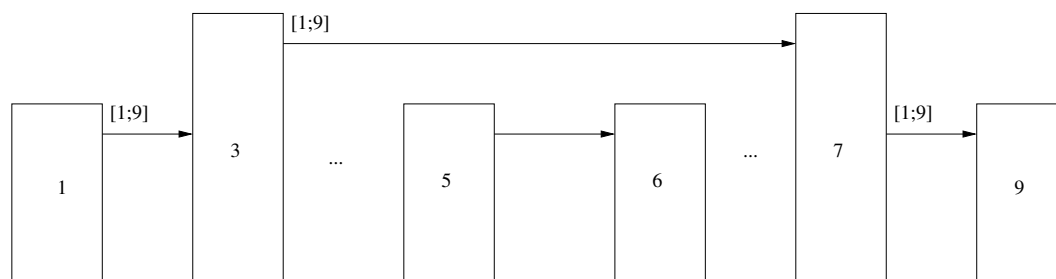


FIG. 3 – Exemple de liste à enjambements adaptée à la recherche d'intervalles, contenant les intervalles $\{[1; 9], [3; 7], [5; 6]\}$.

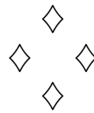
Génération aléatoire de villes. Nous allons considérer la suite (A_n) d'intervalles d'altitudes des villes suivantes. Pour cela nous utilisons les suites (m_n) et (M_n) définies comme suit : $m_n = \min(w_{2n}, w_{(2n+1)})$ et $M_n = \max(w_{2n}, w_{(2n+1)})$.

$$A_n = [m_n; M_n]$$

Insérer les valeurs $\{A_0, \dots, A_{50\,000}\}$ dans la structure de données.

Question 7 Utilisez votre algorithme de recherche pour calculer combien de villes seraient impactées par une montée des eaux aux altitudes suivantes : **a)** w_{10} **b)** $w_{1\,000}$ **c)** $w_{100\,000}$

Question 8 Afin de vérifier l'efficacité de votre algorithme, nous demandons maintenant de compter la somme des nombres de villes impactées respectivement par les altitudes : **a)** $\{w_0, \dots, w_{10}\}$ **b)** $\{w_0, \dots, w_{1\,000}\}$ **c)** $\{w_0, \dots, w_{10\,000}\}$



Le Réchauffement Climatique

Nom, prénom, u₀:

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)

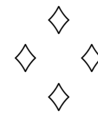
c)

Question 8

a)

b)

c)



Fiche d'évaluation: Le Réchauffement Climatique

Nom, prénom, u₀:

Question 1 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 2 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 3 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 4 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 5 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 6 0-1-2-3-4-5-6-7-8

- a) 0|-----|-----|-----|-----|4
- b) 0|-----|-----|-----|-----|4
- c) 0|-----|-----|-----|-----|4

Question 7 0-1-2-3-4-5-6-7-8



a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

Question 8 0-1-2-3-4-5-6-7-8



a) 0 |-----| 4

b) 0 |-----| 4

c) 0 |-----| 4

