Chapitre 1

Introduction et Rappels

1.1 Qu'est-ce que la géométrie algorithmique ?

La géométrie algorithmique est une discipline apparue dans les années 1970 traitant de problèmes géométriques sous l'angle algorithmique. La géométrie algorithmique tire ses références et entretien des liens étroits avec de nombreux domaines tels que : les mathématiques discrètes et combinatoires, la théorie des polytopes, la recherche opérationnelle, la théorie des graphes et l'informatique en général.

Les applications sont nombreuses :
- Infographie (détections des collisions d'objets, faces cachées, visibilité, reconstruction, ...)
- Robotique (planification de trajectoires)
- Système d'informations géographiques (meilleurs chemins, co-raffinement de cartes, ...)
- CAO/MAO (assemblage, déplacements d'outils, ...)

La géométrie algorithmique est bien représentée par un certain nombre de problèmes fondamentaux :
- Programmation linéaire, polytopes, arrangements
  - Enveloppes convexes
  - Triangulation de Delaunay et diagramme de Voronoi
- Recherche (simpliciale, orthogonale) et localisation
- Triangulations
- Plus courts chemins

Un des développements récents concerne la robustesse et l'implémentation des algorithmes (CGAL, LEDA).

Références :

- Computational Geometry. Algorithms and applications. M. de Berg, O. Cheong, M. van Kreveld, M. Overmars and Schwarzkopf. Springer 2008. (Une très bonne introduction).

- Computational Geometry : An Introduction Through Randomized Algorithms. Ketan Mulmuley. Prentice-Hall, 1994. (Couvre les aspects liés à la randomisation ou à l'analyse randomisée des algorithmes).

- Géométrie Algorithmique. Jean-Daniel Boissonnat et Mariette Yvinec. Ediscience International, 1995.

- Computational Geometry. F. Preparata and M. Shamos. Springer 1985. (Un des premiers ouvrages sur le sujet, un peu daté).

- Handbook of Discrete and Computational Geometry. Edited by Goodman and O'Rourke. CRC PRESS, 1997. (L'état de l'art. Complet et pointu. Idéal pour chercher un résultat).

- Handbook of Computational Geometry. Edited by J. R. Sack and J. Urrutia North Holland, 2000

- The Computational Geometry Impact Task Force Report, B. Chazelle and 36 co-authors, Advances in Discrete and Computational Geometry, Contemporary Mathematics, 223, AMS, Providence, 1999, pp. 407–463.
`http://www.cs.princeton.edu/~chazelle/pubs.html`

1.2 Algorithmes et Complexité

La complexité d'un algorithme fait référence à un modèle de calcul ou plus précisément à un type de calculateur théorique auquel est associé une fonction de coût pour chacune de ses opérations élémentaires. La *complexité* d'un algorithme exprime le rapport asymptotique entre la taille des données d'un problème et le temps (ou l'espace) nécessaire pour exécuter l'algorithme correspondant sur le type de calculateur choisi. Le modèle RAM (Random Access Machine) est un des plus courants en géométrie algorithmique. Il se compose

– d'une suite finie de registres d'entrée pouvant contenir des entiers arbitraires et ne pouvant être accédés qu'en lecture seule,
– d'une bande infinie de registres, indexés par $\mathbb{N}$ et pouvant contenir des entiers arbitraires ; il s'agit intuitivement de la mémoire de la machine ; le registre 0, appelé *accumulateur*, permet d'effectuer des opération arithmétiques,
– d'un programme constitué d'une suite finie d'instructions élémentaires choisies parmi un ensemble prédéfini d'instructions (READ, STORE, ADD, ...) paramétrées par des indices (adressage direct ou indirect) de registres (d'entrée ou non),
– d'un compteur d'instructions contenant l'indice de la prochaine instruction à effectuer,
– (optionnellement) d'une suite finie de registres de sortie. On peut aussi décider que le second registre de la mémoire contient la sortie.

En général on associe un coût unité à chaque instruction élémentaire bien que cette instruction accède à des registres d'indices arbitraires (d'où le nom de RAM). Mais on peut également choisir pour l'accès à un à registre donné un coût logarithmique fonction de son indice. On dira que la complexité d'un programme est $f(n)$ si pour toute entrée de taille n, codée sur n registres, le coût total de l'exécution du programme est au plus $f(n)$. Il s'agit donc de la complexité dans le cas le pire relativement à tous les jeux de données de taille n.

On suppose en général que la taille des registres est constante ce qui revient à définir la taille de l'entrée par le nombre total de bits utilisés pour la coder. Dans ce cas, si on restreint les opérations arithmétiques à l'addition et la soustraction, bien que le modèle

RAM puisse manipuler en temps constant des entiers de tailles arbitraires et accéder à un registre de mémoire en temps constant, il est polynomialement équivalent au modèle de la machine de Turing (MT) ! C'est-à-dire que toute fonction calculable par une RAM peut être calculée par une MT en temps dépendant de manière polynomiale du temps requis par la RAM, et réciproquement.

Pour se concentrer sur la partie purement combinatoire d'un problème, on peut supposer que la machine RAM peut stocker et manipuler (effectuer les opérations/comparaisons $+, -, *, /, <, =, \sqrt{}, \ldots$) des réels en temps constant, on parle alors de modèle Real-RAM. Ce pourra être le cas lorsqu'on doit manipuler des coordonnées de points dans $\mathbb{R}^n$ ou d'autres objets géométriques. Les registres pouvant contenir des réels sont alors différenciés de ceux qui peuvent contenir des addresses. Dans le cas contraire le modèle de machine serait bien trop puissant. On pourra consulter [vEB90] pour une sensibilisation aux questions de complexités et de modèles de calculs. *Sauf mention explicite d'un autre modèle, tous les calculs de complexités dans ce cours utilisent le modèle de la Real-RAM.*

Un certain vocabulaire, plus ou moins propre à la géométrie algorithmique, permet de décrire les algorithmes en fonction de leur adaptation au flux des données, ou en fonction de leur principe de fonctionnement. Ainsi un algorithme est dit *statique* ou *hors-ligne* s'il s'applique à un jeu de données connu avant son exécution. On parle d'algorithme *en ligne* ou *semi-dynamique* si au contraire de nouvelles données peuvent être ajoutées, et que l'algorithme est capable, au cours du temps, de maintenir la solution du problème associé à toutes les données accumulées. Enfin, un algorithme *dynamique* permet d'ajouter ou de supprimer des données au cours du temps.

La conception d'un algorithme peut être *déterministe* ou *randomisée* selon que l'algorithme aura ou non toujours le même comportement face à un même jeu de données. Dans ce dernier cas, les variations de comportement sont liées à l'usage de générateurs aléatoires de nombres (ou bits) qui permettent de modifier le déroulement de l'algorithme.

On peut analyser de manière randomisée la complexité d'un l'algorithme en la considérant comme une variable aléatoire. Cette variable aléatoire peut dépendre soit des générateurs aléatoires dans le cas des algorithmes randomisés, soit d'une distribution a priori sur les données dans le cas déterministe. Dans le cas d'un algorithme randomisé, on suppose généralement que le jeu de données est le pire possible. Ainsi lorsqu'on calcule l'espérance de la complexité, encore appelée la *complexité en moyenne*, on se place dans le cas où les données maximise cette espérance. On peut également s'intéresser à la *queue de distribution*, c'est à dire à la probabilité que la complexité dépasse une certaine valeur.

Les méthodes *incrémentales, par balayage ou diviser pour régner* apparaissent de manière récurrente en géométrie algorithmique comme principe de résolution de problèmes.

Références :

- Calculabilité, complexité et approximation. J.-F. Rey. Vuibert, 2004.

- The Design an Analysis of Computer Algorithm. Chap. 1. Aho, Hopcroft et Ullman. Addison Wesley, 1974.

- Computational complexity. Christos Papadimitriou. Addison Wesley, 1995.

- Machine Models and Simulations. van Emde Boas, P. in Handbook of theoretical computer science, vol A. Elsevier, 1990.

- What is a "pointer machine" ?. M. Ben-Amram. SIGACT News, 26 (1995), 88–95.

1.3 Tri

Le tri d'éléments, pris dans un univers totalement ordonné, est un des problèmes basiques et essentiels de l'algorithmique. Étant donné une séquence de N éléments le problème est de trier cette séquence, i.e. de trouver la permutation transformant cette séquence en la séquence ordonnée de ses éléments.

De très nombreux algorithmes permettent de trier. Il y en a de plus "efficaces" que d'autres. On mesure cette efficacité par le nombre (maximal, moyen, ...) d'opérations élémentaires utilisées pour trier une suite quelconque de taille N, c.a.d. par la complexité de l'algorithme considéré. L'opération élémentaire pour les tris est la comparaison.

Exemples :
- **tri naïf** : complexité ?
- **tri par insertion dichotomique** : Insérer par dichotomie le (k + 1)-ième élément dans la liste triée des k premiers éléments. On note $C(k)$ la complexité dans le pire des cas, en anglais on parle de worst case analysis, de l'insertion du (k + 1)-ième élément. On a

$$C(0) = 0$$

$$C(k) = 1 + C(\lceil \frac{k-1}{2} \rceil) = 1 + C(\lfloor \frac{k}{2} \rfloor)$$

  On vérifie par récurrence que

$$C(k) = \lceil 1 + \log k \rceil$$

  en utilisant le fait que pour k > 1

$$\lceil \log 2 \lfloor \frac{k}{2} \rfloor \rceil = \lceil \log k \rceil.$$

  La complexité B(N) du tri de N éléments est donc

$$B(N) = \sum_{k=0}^{N-1} C(k) = O(N \log N).$$

  Attention, cette analyse ne tient pas compte de la gestion de la mémoire.
- **tri fusion** : On scinde en deux la liste à trier et on trie récursivement chacune des sous-listes avant de les fusionner.

$$CF(N) = CF(\lceil N/2 \rceil) + CF(\lfloor N/2 \rfloor) + N = O(N \log N).$$

- **Quicksort (Hoare 1962)** : Choisir aléatoirement un pivot dans la liste à trier puis scinder en deux la liste suivant les éléments plus petits ou plus grands que le pivot. Trier récursivement chaque sous-liste.

Il s'agit du tri utilisé par UNIX. Très efficace en pratique.
Analyse dans le cas le pire : Chaque clé est choisie une unique fois comme pivot. Il y a au plus N-1 comparaisons à effectuer avec un pivot, d'où une complexité en $O(N^2)$. Ce sera le cas si les pivots sont pris dans l'ordre croissant des clés.
Analyse en moyenne : les clés $K_1, \ldots, K_N$ étant indexées selon leur ordre de grandeur, on note $X_{i,j}$ la variable aléatoire qui vaut 1 si K_i et K_j sont comparées et 0 sinon. K_i et K_j sont comparées si et seulement si elles n'ont pas été séparées avant que l'une soit choisie comme pivot, c.a.d. si et seulement si l'une de ces deux clés est choisie comme pivot en premier parmi les $j-i+1$ clés comprises (au sens large) entre K_i et K_j. Ceci se produit avec une probabilité $2/(j-i+1)$, d'où

$$E(\sum_{i<j} X_{i,j}) = \sum_{i<j} \frac{2}{j-i+1} = \sum_{j=2}^{N} \sum_{k=2}^{j} \frac{2}{k} \leq 2NH_N = O(N \log N)$$

où $H_N = \sum_{i=1}^{N} 1/i$ est le N-ième nombre harmonique.
Voir également pour une preuve, l'analyse en moyenne de la hauteur d'un arbre de recherche aléatoire.

Le tri d'entiers codés en binaire peut utiliser d'autres moyens que la comparaison. En faisant des hypothèses sur la taille des nombres à trier, on obtient des complexités moindre. Pour un article récent sur le sujet, voir :

- Integer Sorting in $O(n\sqrt{\log \log n})$ Expected Time and Linear Space. Y. Han and M. Thorup. pp 135 - 144. FOCS 2002, Vancouver, Canada.

1.4 Arbres binaires

L'ensemble des arbres binaires est défini inductivement par l'équation ensembliste $\mathcal{B} = \square + (\circ, \mathcal{B}, \mathcal{B})$. Dit autrement un arbre binaire est soit l'arbre vide soit obtenu en accrochant deux arbres binaires (sous-arbres gauche et droit) à une racine. On note $sag(B)$ (resp. $sad(B)$) le sous-arbre gauche (resp. droit) d'un arbre binaire B.

Une opération importante sur les arbres binaires est la *rotation* :

$$(\circ, (\circ, b_1, b_2), b_3) \mapsto (\circ, b_1, (\circ, b_2, b_3))$$

Codage informatique : un noeud est représenté par un objet à trois champs : la clé du noeud, deux pointeurs sur des noeuds représentants les sous-arbres gauche et droit.

Définitions : *Racine, noeud, noeud interne, noeud externe ou feuille, arête, enfant, parent*, etc...

Un arbre binaire *complet* est un arbre binaire dont tous les noeuds internes ont deux enfants. On peut compléter un arbre binaire quelconque en ajoutant deux (resp. une) feuilles à tous ses noeuds ayant zéro (resp. un) enfant.

Propriété : Dans un arbre binaire complet on a :
#{noeuds externes} = #{noeuds internes} + 1.

Preuve : Orienter les arêtes des noeuds parents vers les enfants. Le nombre d'arêtes sortantes est le double du nombre de noeuds internes, c'est aussi le nombre d'arêtes entrantes qui est le nombre total de noeuds moins 1 (pour la racine). □

On définit récursivement les *taille, hauteur, longueurs de cheminement interne et externe* d'un arbre binaire complet par

taille, hauteur, lci, lce(□) = 0
taille(b) = 1 + taille(sag(b)) + taille(sad(b))
hauteur(b) = 1 + max{ hauteur(sag(b)), hauteur(sad(b)) }
lci(b) = taille(b) - 1 + lci(sag(b)) + lci(sad(b))
lce(b) = taille(b) + 1 + lce(sag(b)) + lce(sad(b))

On définit également récursivement la profondeur d'un noeud par
profondeur(racine) = 0
profondeur(noeud) = 1 + profondeur(parent(noeud))
C'est aussi le nombre d'arêtes du chemin simple reliant le noeud à la racine.

Dit autrement un arbre binaire a une taille égale à son nombre de noeuds internes, une hauteur égale à la profondeur maximale d'un noeud externe et une longueur de cheminement interne (resp. externe) égale à la somme des profondeurs de tous ses noeuds internes (resp. externes).

On vérifie par récurrence que lce(b) = lci(b) + 2 taille(b).

Un arbre binaire complet est dit *parfait* si tous ses noeuds externes ont la même profondeur.

Propriétés : Dans un arbre binaire parfait, b, de hauteur h on a :

$$\text{taille(b)} = \sum_{i=0}^{h-1} 2^i = 2^h - 1,$$

$$\text{lce(b)} = h2^h,$$
$$\text{lci(b)} = (h-2)2^h + 2.$$

Propriétés : Le nombre d'arbres binaires à n sommets est le *nombre de Catalan* $C_n = \frac{1}{n+1}\binom{2n}{n}$.

1.5 Borne inférieure pour la complexité moyenne des tris

Les différentes comparaisons effectuées lors du déroulement d'un algorithme *déterministe* de tri sur N clés données s'organisent selon un arbre binaire complet. Les feuilles de cet arbre correspondent aux $N!$ permutations possibles des N clés. Le coût maximal (resp. moyen) de cet algorithme est la profondeur maximale (resp. moyenne) des feuilles de son arbre de comparaisons. C'est encore, d'après la section précédente, la hauteur (resp. le rapport de la lce par la taille + 1) de son arbre de comparaisons.

Propriété : pour une taille, t, fixée la lce d'un arbre binaire complet est minimisée par l'arbre parfait.

Preuve : On note b_t l'arbre parfait de taille t et b un arbre quelconque de taille t. De la formule récursive de la lce, on tire
lce(b) - lce(b_t) = lce(sag(b)) + lce(sad(b)) - 2 lce($b_{(t-1)/2}$)
En utilisant l'hypothèse de récurrence lce(b) $\geq$ lce(b_t) $= (t+1)\log(t+1)$ aux ordres convenables on a

$$\text{lce}(b) - \text{lce}(b_t) \geq (t_g+1)\log(t_g+1) + (t_d+1)\log(t_d+1) - (t+1)\log(t+1)$$

avec $t_g + t_d = t - 1$. On vérifie, en utilisant par exemple la convexité de $x \mapsto x\log x$, que le membre de droite est positif ou nul. La formule étant triviale pour un arbre de taille 0, ceci permet de confirmer l'hypothèse de récurrence.

De la propriété précédente on déduit que la complexité moyenne d'un algorithme de tri est minorée par $N!\log(N!)/N! = O(N\log N)$ □

1.6 Coefficients du binôme

Il existe une quantité impressionnante de formules à propos de ces coefficients. Elles interviennent souvent en combinatoire, mathématique discrète, géométrie algorithmique, etc... dès que l'on cherche à estimer dans un ensemble le nombre de sous-ensembles vérifiant telle ou telle propriété. On pourra consulter le chapitre 5 de

Mathématiques concrètes - Fondations pour l'informatique. R.L. Graham, D. E. Knuth, O. Patashnik. Vuibert 2003, 2e édition.

Définition 1.1 *Un* échantillon *de taille k d'un ensemble est un sous-ensemble de cardinal k. Le nombre d'échantillons de taille k dans un ensemble de taille n est noté C_n^k (lire "binomiale de n, k") ou $\binom{n}{k}$.*

On note N_n un ensemble à n éléments.

1.6.1 Formulaire I - Identités

- $\binom{n}{k} = \binom{n}{n-k}$.

Preuve : L'application qui envoie un échantillon de taille k sur son complémentaire (de taille $n-k$) est une bijection.

- **Formule de Pascal** $\binom{n+1}{k+1} = \binom{n}{k+1} + \binom{n}{k}$.

Preuve : Compter séparément les échantillons contenant ou non un élément donné.

- $\binom{n}{k}\binom{k}{p} = \binom{n}{p}\binom{n-p}{k-p}$.

Preuve : Compter de deux manières différentes le nombre d'échantillons de taille k dans N_n dont p éléments sont distingués.

Pour $p = 1$ on obtient :

- $k\binom{n}{k} = n\binom{n-1}{k-1}$ puis, avec la formule de Pascal, $\binom{n}{k+1} = \frac{n-k}{k+1}\binom{n}{k}$ et $\binom{n}{k} = \frac{n}{n-k}\binom{n-1}{k}$

On en déduit par récurrence :

- $\binom{n}{k} = \prod_{i=0}^{k-1} \frac{n-i}{k-i}$.

- **Formule du binôme** $(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}$.

Preuve : Par récurrence avec la formule de Pascal ou en exprimant comment obtenir le nombre de monômes $x^k y^{n-k}$.

- $\sum_{k=0}^{n} \binom{n}{k} = 2^n$.

Preuve : Compter de deux manières différentes le nombre total de sous-ensembles de N_n.

- $\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n}$.

Preuve : Écrire $\binom{n}{k}^2 = \binom{n}{k}\binom{n}{n-k}$ et remarquer que l'ensemble des paires d'échantillons, (E_k, E'_{n-k}), de taille respective k et $n-k$ pris dans deux ensembles E et E' à n éléments est en bijection avec l'ensemble des échantillons de taille n de $E \cup E'$.

- $\sum_{i=k}^{n} \binom{i}{k} = \binom{n+1}{k+1}$.

Preuve : Numéroter de 1 à $n+1$ les éléments de N_{n+1}, puis partitionner l'ensemble de ses échantillons de taille $k+1$ en fonction de leur plus grand élément.

- $\sum_{i=1}^{n} \frac{1}{i}\binom{i}{k} = \frac{1}{k}\binom{n}{k}$.

Preuve : Écrire $\frac{1}{i}\binom{i}{k} = \frac{1}{k}\binom{i-1}{k-1}$ et utiliser la formule précédente.

- $\sum_{i=0}^{n} i\binom{i}{k} = (k+1)\binom{n+2}{k+2} - \binom{n+1}{k+1}$.

Preuve : Écrire $i\binom{i}{k} = (i+1)\binom{i}{k} - \binom{i}{k} = (k+1)\binom{i+1}{k+1} - \binom{i}{k}$.

1.6.2 Formulaire II - Estimations

- $\binom{n}{k} \leq n^k$.

Preuve : Il y a moins d'échantillons de taille k dans N_n que d'applications de N_k vers N_n. (Écrire $\binom{n}{k} = \binom{n}{n-k}$ si $k > n/2$).

- $(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{en}{k})^k$ (et même $\sum_{i=0}^{k} \binom{n}{i} \leq (\frac{en}{k})^k$).

Preuve : Pour l'inégalité de gauche écrire $\binom{n}{k} = \prod_{i=0}^{k-1} \frac{n-i}{k-i}$ et remarquer que $\frac{n-i}{k-i} \geq \frac{n}{k}$. Pour l'inégalité de droite, écrire : $\forall x > 0 : \exp(nx) \geq (1+x)^n = \sum_{i=0}^{n} \binom{n}{i} x^i \geq \binom{n}{k} x^k$. D'où $\binom{n}{k} \leq x^{-k} \exp(nx)$ et on conclut en posant $x = k/n$.

- $$\binom{n}{\lfloor n/2 \rfloor} = \binom{n}{\lceil n/2 \rceil} = \max_{0 \leq k \leq n} \binom{n}{k}.$$

Preuve : De $\binom{n}{k} = \frac{n-k+1}{k}\binom{n}{k-1}$ on tire $\binom{n}{k} > \binom{n}{k-1}$ si $k \leq n/2$, et on utilise l'égalité $\binom{n}{k} = \binom{n}{n-k}$ pour $k \geq n/2$.

- $$\frac{2^{2n}}{2\sqrt{n}} \leq \binom{2n}{n} \leq \frac{2^{2n}}{\sqrt{2n}}.$$

Preuve : $(\binom{2n}{n})^2 = \frac{((2n)!)^2}{(n!)^4} = \frac{(1.2...2n)^2}{(1.2...n)^4} = 2^{4n}\frac{(1.2...2n)^2}{(2.4...2n)^4} = 2^{4n}\frac{(1.3...(2n-1))^2}{(2.4...2n)^2}$. D'où $(\binom{2n}{n})^2 = \frac{3^2}{2.4}\frac{5^2}{4.6}\cdots\frac{(2n-1)^2}{(2n-2)2n}\frac{2^{4n}}{2.2n} \geq \frac{2^{4n}}{2.2n}$, et $(\binom{2n}{n})^2 = \frac{1.3}{2^2}\frac{3.5}{4^2}\cdots\frac{(2n-1)(2n+1)}{2n^2}\frac{2^{4n}}{2n+1} \leq \frac{2^{4n}}{2n}$

1.7 Probabilité discrète élémentaire

Espace de probabilité, variable aléatoire, indépendance, espérance, inégalité de Markov, probabilité conditionnelle, inégalité de Chebyshev, inégalité de Jensen, distribution géométrique, binomiale. Bornes de Chernoff.

Je vais suivre le 'reading assignment' du 'graduate program' de Zurich :

http ://www.ti.inf.ethz.ch/ew/courses/RandAlgs00/RandAlgs.html

Autres références :
- The probalistic Method. Alon and Spencer, John Wiley 2000.
- http ://kam.mff.cuni.cz/∼matousek/lectnotes.html
- http ://cermics.enpc.fr/∼delmas/enseignement.html

1.7.1 Définitions et propriétés élémentaires

Définition 1.2 *Un* espace de probabilité *est un ensemble Ω munit d'une application $P : \Omega \to \mathbb{R}^+$ telle que $\sum_{\omega \in \Omega} P(\omega) = 1$ (lorsque Ω est infini cette somme est définie comme le sup. sur les parties finies). Un élément de Ω est appelé une* réalisation *et une partie de Ω est appelée un* événement.

Définition 1.3 *Une* variable aléatoire *(réelle) est une application (à valeurs réelles) définie sur un espace de probabilité.*

Définition 1.4 *Deux variables aléatoires X et Y sont dites* indépendantes *si*

$$\forall x \in ImX, \forall y \in ImY : \; P(X = x \wedge Y = y) = P(X = x).P(Y = y)$$

Les n variables aléatoires $X_i, i = 1, \ldots, n$ sont mutuellement indépendantes *si pour tout $(x_1, x_2, \ldots, x_n) \in ImX_1 \times ImX_2 \times \ldots \times ImX_n$:*

$$P(X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots \wedge X_n = x_n) = P(X_1 = x_1).P(X_2 = x_2) \ldots P(X_n = x_n)$$

Plus généralement, les variables aléatoires d'une famille $\{X_i\}_{i\in I}$ indéxée par un ensemble dénombrable I sont mutuellement indépendantes *si pour toute partie finie $J \subset I$ les variables de la famille $\{X_j\}_{j\in J}$ sont mutuellement indépendantes.*

Exercice 1.5 *Définir la notion d'indépendance à l'aide des probabilités conditionnelles (cf. définition 1.16).*

Exercice 1.6 *Soient deux variables aléatoires indépendantes $X : \Omega \to F$ et $Y : \Omega \to G$ et une fonction $f : F \to H$. Montrer que $f(X)$ et Y sont indépendantes.*

Définition 1.7 *L'*espérance *d'une variable aléatoire X est*

$$E(X) = \sum_{x\in ImX} xP(X = x) = \sum_{\omega\in\Omega} X(\omega)P(\omega)$$

lorsque ces sommes existent. Sa variance est

$$var(X) = E((X - E(X))^2) = E(X^2) - E(X)^2.$$

Par la suite on considère des variables aléatoires dont l'espérance existe. Le lemme suivant est une application directe de la définition de l'espérance (sous sa seconde forme ci-dessus).

Lemme 1.8 *L'espérance est linéaire.*

Exercice 1.9 *Soient une variable aléatoire $X : \Omega \to F$ et une fonction $f : F \to \mathbb{R}$. Montrer que*

$$E(f(X)) = \sum_{x\in ImX} f(x)P(X = x).$$

Le lemme suivant est particulier aux fonctions entières non négatives mais bien utile.

Lemme 1.10 *Si X est à valeurs naturelles alors*

$$E(X) = \sum_{k\in\mathbb{N}} P(X > k).$$

Preuve : $\displaystyle\sum_{k\in\mathbb{N}} P(X > k) = \sum_{k\in\mathbb{N}} \sum_{i>k} P(X = i) = \sum_{i>0} iP(X = i) = E(X).$ □

Lemme 1.11 *Soient X et Y deux variables aléatoires indépendantes, alors*

$$E(XY) = E(X)E(Y) \text{ et } var(X + Y) = var(X) + var(Y).$$

Preuve :

$$E(XY) = \sum_{\omega\in\Omega} X(\omega)Y(\omega)P(\omega) = \sum_{x,y\in\mathbb{R}} \sum_{X(\omega)=x\wedge Y(\omega)=y} xyP(\omega) =$$

$$\sum_{x,y\in\mathbb{R}} xyP(X(\omega)=x\wedge Y(\omega)=y) = \sum_{x,y\in\mathbb{R}} xyP(X(\omega)=x).P(Y(\omega)=y) = E(X)E(Y).$$

On en déduit l'égalité sur les variances. □

Lemme 1.12 (Inégalité de Markov) *Soit X une variable aléatoire non négative, alors*

$$\forall\lambda > 0 : P(X \geq \lambda) \leq \frac{E(X)}{\lambda}.$$

Il y a égalité si et seulement si pour tout ω de probabilité non nulle : $X(\omega) \in \{0, \lambda\}$.

Preuve :

$$E(X) = \sum_{\omega\in\Omega} X(\omega)P(\omega) \geq \sum_{X(\omega)\geq\lambda} X(\omega)P(\omega) \geq \lambda P(X \geq \lambda).$$

□

Dit autrement, la probabilité qu'une variable aléatoire non négative dépasse un certain nombre de fois son espérance est majorée par l'inverse de ce nombre.

Lemme 1.13 (Inégalité de Chebychev) *Soit X une variable aléatoire non négative, alors*

$$\forall\lambda > 0 : P(|X - E(X)| \geq \lambda) \leq \frac{var(X)}{\lambda^2}.$$

Preuve : Appliquer Markov à $(X - E(X))^2$. □

Lemme 1.14 (Inégalité de Jensen) *Soit X une variable aléatoire et f une fonction convexe, alors*

$$f(E(X)) \leq E(f(X)).$$

Preuve : Si Ω est fini cette inégalité traduit simplement le fait que l'image du barycentre d'un ensemble fini de points par une fonction convexe est majorée par le barycentre des images de ces points. Le cas général demande un peu plus de travail. Je note $\tau_f(x, y) = \frac{f(x)-f(y)}{x-y}$ le taux d'accroissement de f. Pour tout $s \leq u$ on a $\tau_f(E(X), s) \leq \tau_f(E(X), u)$. Soit $\beta = \sup_{s<E(X)} \tau_f(E(X), s)$, on vérifie à l'aide de l'inégalité précédente que

$$\forall x,\ f(x) \geq f(E(X)) + \beta(x - E(X)).$$

Dit autrement, le graphe de f est au dessus de sa "tangente" au point $(E(X), f(E(X)))$. En substituant $X(\omega)$ à x et en prenant les espérances des deux membres de l'inégalité on obtient la relation cherchée. □

1.7.2 Probabilités conditionnelles

Définition 1.15 *Soit (Ω, P) un espace de probabilité et soit $B \subset \Omega$ un événement de probabilité non nulle. L'*espace induit *(B, P_B) est l'espace de probabilité sur l'ensemble B avec $P_B(\omega) = P(\omega)/P(B)$.*

Définition 1.16 *Soient A et B deux événements avec $P(B) > 0$, la* probabilité conditionnelle *de A par rapport à B est définie par*

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}.$$

Soient X une variable aléatoire et B un événement de probabilité positive, alors la variable aléatoire $X|B$ est la restriction de X à B dans l'espace induit (B, P_B).

On vérifie que $P_B(X|B = x) = P(X = x|B)$.

Exercice 1.17 *Soient deux événements $B \subset A$ avec $P(B) > 0$ et X une variable aléatoire. Montrer que $(X|A)|B = X|B$.*

Lemme 1.18 *Soient A un événement, X une variables aléatoire, et $(B_i)_{i\in I}$ une famille d'événements disjoints dont la réunion est l'espace Ω. En particulier, $\sum_{i\in I} P(B_i) = 1$. Alors,*

$$P(A) = \sum_{i\in I} P(A|B_i)P(B_i)$$

et

$$E(X) = \sum_{i\in I} E(X|B_i)P(B_i).$$

Preuve : Par définition de la probabilité conditionnelle :

$$\sum_{i\in I} P(A|B_i)P(B_i) = \sum_{i\in I} P(A \wedge B_i)$$

D'où la première égalité par hypothèse sur les B_i. Pour la seconde égalité, on écrit :

$$\sum_{i\in I} E(X|B_i)P(B_i) = \sum_{i\in I}\sum_{x\in\mathbb{R}} xP_{B_i}(X|B_i = x)P(B_i) = \sum_{x\in\mathbb{R}}\sum_{i\in I} xP(X = x|B_i)P(B_i)$$

et on termine à l'aide de la première égalité. □

Remarque : Ce lemme est souvent bien pratique pour évaluer ou majorer une espérance ou une probabilité. En effet, si $E(X|B_i)$ (resp. $P(A|B_i)$) est constant ou uniformément borné par rapport aux B_i alors cette constante ou borne reste valable *inconditionnellement*, i.e. pour $E(X)$ (resp. $P(A)$) : il suffit de mettre en facteur la constante ou borne dans la seconde égalité et d'utiliser le fait que $\sum_{i\in I} P(B_i) = 1$.

Exercice 1.19 *Si X est une variable aléatoire indépendante de l'événement B (i.e. $P_B(X|B = x) = P(X = x)$), montrer que $E(X|B) = E(X)$.*

Exercice 1.20 *Soient $A_1, A_2, \ldots, A_n$ et B des événements. Montrer que*

$$P(\bigwedge_i A_i|B) = \prod_i P(A_i|\bigwedge_{j>i} A_j \wedge B)$$

Exercice 1.21 *Soient A un événement et X une variables aléatoire. Montrer une version conditionnelle du lemme 1.18, i.e.*

$$\begin{aligned} P(A|B) &= \sum_i P(A|B_i)P(B_i|B) \\ E(X|B) &= \sum_i E(X|B_i)P(B_i|B) \end{aligned}$$

où B est la réunion disjointe des B_i.

Exercice 1.22 *Soit (Ω, P) un espace de probabilité, U un ensemble fini et $A : \Omega \to \mathcal{P}(U)$. Montrer que*

$$E(|A|) = \sum_{u \in U} P(u \in A)$$

Exercice 1.23 *Sous les hypothèses de l'exercice précédent, on considère de plus une famille de variables aléatoires $(X_u)_{u \in U}$ telle que l'espérance conditionnelle $E(X_u\,|u \in A\,)$ est uniformément bornée par une constante c. Montrer que*

$$E(\sum_{u \in A} X_u) \leq cE(|A|)$$

où $\sum_{u \in A} X_u$ désigne la variable aléatoire $\omega \mapsto \sum_{u \in A(\omega)} X_u(\omega)$. On notera en particulier que la linéarité de l'espérance ne peut s'appliquer à une somme portant sur un ensemble qui dépend de la réalisation ω.

Montrer par un contre-exemple que l'inégalité ci-dessus est généralement fausse si on suppose seulement que $E(X_u)$ est uniformément bornée par c.

1.7.3 Lois classiques

Définition 1.24 (loi de Bernoulli) *Supposons avoir un sac avec une proportion p de boules blanches et $1-p$ de boules rouges. Si on tire une boule au hasard dans le sac, alors la probabilité d'obtenir une boule blanche est p et la probabilité d'obtenir une boule rouge est $1 - p$. On dit que la variable aléatoire valant 1 lorsque la boule tirée est blanche et 0 sinon suit une* loi de Bernoulli de paramètre p.

Définition 1.25 (loi géométrique) *Avec les hypothèses précédentes la probabilité d'obtenir une boule blanche après i tirages avec remise vaut $(1-p)^{i-1}p$. Si X est la variable aléatoire valant le nombre de tirages effectués avant d'obtenir une boule blanche alors cette probabilité est précisément $P(X=i)$ et on dit que X a une* distribution (ou loi) géométrique de paramètre p.

On vérifie par calcul direct que $E(X) = 1/p$ et $var(X) = (1-p)/p^2$. Si b_1 est la couleur de la première boule tirée, on peut aussi écrire $E(X) = E(X|b_1 = \text{ blanc})p + E(X|b_1 = \text{ rouge})(1-p)$, puis remarquer que $(X|b_1 = \text{ blanc}) = 1$ et que $(X|b_1 = \text{ rouge}) = 1+Y$, où Y est le nombre de tirages effectués avant d'obtenir une boule blanche à partir du second tirage. Évidement, Y a la même distribution que X et on en déduit une équation simple pour $E(X)$. Un calcul similaire permet de calculer 'directement' $var(X)$.

Question : Quel est l'espace de probabilités de X ? Votre modèle entre-t-il dans le cadre des probabilités discrètes ? (cf. notion de schéma de Bernoulli).

Définition 1.26 (loi binomiale) *Toujours avec les mêmes hypothèses, on considère la variable aléatoire Y valant le nombre de boules blanches obtenues après n tirages avec remise. On dit que Y suit une* distribution binomiale de paramètres n et p. *On a clairement $P(Y=i) = \binom{n}{i}p^i(1-p)^{n-i}$.*

Remarque : si Y_i est la variable aléatoire (de Bernoulli) qui vaut 1 si le i-éme tirage est une boule blanche et 0 sinon, alors $Y = \sum_{1\leq i\leq n} Y_i$.

On en déduit que $E(Y) = \sum_{1\leq i\leq n} E(Y_i) = np$. De plus les Y_i étant indépendantes, on a $var(Y) = \sum_{1\leq i\leq n} var(Y_i) = np(1-p)$.

Question : Quel est l'espace de probabilité de Y ?

Définition 1.27 (loi binomiale négative) *Toujours avec les mêmes hypothèses on considère la variable aléatoire Z valant le nombre de tirages nécessaires pour obtenir n boules blanches. On dit que Z a une* distribution binomiale négative de paramètres n et p. *On a $P(Z=i) = \binom{i-1}{n-1}p^n(1-p)^{i-n}$.*

Remarque : si Z_i est la variable aléatoire qui vaut le nombre de tirages entre les tirages des i-ème et $(i+1)$-ème boules blanches, alors $Z = \sum_{0\leq i\leq n-1} Z_i$. Les Z_i sont indépendantes et suivent une distribution géométrique de paramètre p.

On en déduit que $E(Z) = \sum_{0\leq i\leq n-1} E(Z_i) = \frac{n}{p}$. (On peut aussi faire un calcul direct en utilisant le fait que $\sum_i \binom{i+n}{n}(1-p)^i = \frac{1}{n!}\sum_i (i+n)\dots(i+1)(1-p)^i = \frac{1}{n!}(-1)^n(\frac{1}{p})^{(n)} = \frac{n!}{p^{n+1}}$). On a également $var(Z) = \sum_{0\leq i\leq n-1} var(Z_i) = \frac{n(1-p)}{p^2}$.

1.7.4 Technique de Chernoff

Soit X une variable aléatoire égale à la somme $\sum_{1\leq i\leq n} X_i$ de n variables aléatoires indépendantes et de lois identiques (i.i.d). La technique de Chernoff [Che52] permet en

général de trouver de bons majorants pour $P(X \geq x)$ où x est choisit comme un écart à la valeur moyenne $E(X)$ de X. Pour cela on considère un réel $\lambda > 0$ et on écrit

$$P(X \geq x) = P(e^{\lambda X} \geq e^{\lambda x}) \leq E(e^{\lambda X})/e^{\lambda x} = \prod_{1 \leq i \leq n} E(e^{\lambda X_i})/e^{\lambda x} = E(e^{\lambda X_1})^n/e^{\lambda x}.$$

La première inégalité est celle de Markov, la seconde égalité provient de l'indépendance des variables et la dernière de l'identité des lois des X_i. Il reste à choisir convenablement λ pour obtenir une bonne majoration.

Exemples d'application de la technique de Chernoff.

Lemme 1.28 *Soit une variable aléatoire $X = \sum_{1 \leq i \leq n} X_i$ où les X_i sont mutuellement indépendantes à valeurs dans $\{-1, 1\}$ avec $P(X_i = 1) = P(X_i = -1) = 1/2$. Alors*

$$\forall x > 0 : \qquad P(X \geq x) < e^{-\frac{x^2}{2n}}$$

Preuve : Par la technique de Chernoff on a pour tout $\lambda > 0 : P(X \geq x) \leq E(e^{\lambda X_1})^n/e^{\lambda x} = (\cosh \lambda)^n/e^{\lambda x}$. En développant cosh en série entière on vérifie que $\cosh \lambda < e^{\lambda^2/2}$, d'où $P(X \geq x) < e^{n\lambda^2/2 - \lambda x}$. On obtient le résultat en choisissant $\lambda = x/n$. □

Lemme 1.29 *Soit une variable aléatoire X de loi binomiale négative de paramètres n et $1/2$. Alors*

$$\forall x \geq 3 : \qquad P(X \geq (2+x)n) < \exp(-nx/4)$$

Preuve : Notons que $E(X) = 2n$. Par la technique de Chernoff on a pour tout $\lambda > 0$: $P(X \geq (2+x)n) \leq E(e^{\lambda X_1})^n/e^{\lambda(2+x)n}$ où X_1 suit une loi géométrique de paramètre $1/2$. Or

$$E(e^{\lambda X_1}) = \sum_{i=1}^{\infty} e^{\lambda i}/2^i = \frac{e^\lambda}{2 - e^\lambda}.$$

Cette dernière égalité supposant $e^\lambda < 2$. On a dans ce cas $P(X \geq (2+x)n) \leq (\frac{e^{-\lambda(1+x)}}{2-e^\lambda})^n$. On choisit λ tel que $e^\lambda = 1 + \frac{x}{2+x}$ (donc $e^\lambda < 2$), d'où, en utilisant l'inégalité $1 - u < e^{-u}$ pour $u > 0$:

$$\frac{e^{-\lambda(1+x)}}{2 - e^\lambda} = (1 - \frac{x}{2+2x})^{1+x}(1 + x/2) < e^{-\frac{x}{2+2x}(1+x)}(1 + x/2) = e^{-x/2}(1 + x/2)$$

Or pour $x \geq 3$ on vérifie que $1 + x/2 < e^{x/4}$, ce qui permet de conclure. □

Références :

- Computational Geometry. An Introduction Trough Randomized Algorithms. K. Mulmuley, Prentice Hall, 1994.

1.8 Exemples d'applications de la méthode probabiliste

1.8.1 Analyse arrière

On munit l'ensemble des permutations de [1,n] de la distribution uniforme. On considère la variable aléatoire X comptant le nombre de minima successifs en lisant une permutation $(a_1, \ldots, a_n)$ de gauche à droite.

$$X = |\{i \in [1,n] \,|\, a_i = \min\{a_1, \ldots, a_i\}\}|$$

On considère également la variable aléatoire X_i valant 1 si a_i est un minimum et 0 sinon. Alors $X = \sum_{1 \leq i \leq n} X_i$. Pour calculer $P(X_i = 1)$, i.e. $P(a_i = \min\{a_1, \ldots, a_i\})$, on fixe $a_{i+1}, \ldots, a_n$, d'où le nom d'analyse arrière. On remarque alors que

$$P(X_i = 1 \,|\, a_{i+1}, \ldots, a_n \text{ fixés}) = 1/i$$

d'où $P(X_i = 1) = 1/i$ (cf. lemme 1.18).
On en déduit $E(X) = \sum_{1 \leq i \leq n} 1/i = H_n$.

1.8.2 Nombres harmoniques

On vérifie que $\forall n \geq 1 \qquad \ln(n+1) \leq H_n \leq 1 + \ln n$.

1.8.3 Ensembles indépendants

Un sous-ensemble de sommets d'un graphe est *indépendant* si le graphe induit sur ces sommets ne contient pas d'arête.

Théorème 1.30 *Tout graphe G à n sommets et m arêtes contient un sous-ensemble de sommets indépendants de taille au moins $\lfloor n/\sqrt{m} \rfloor$.*

Preuve : On regarde la probabilité pour qu'un sous-ensemble de k sommets soit indépendant. On pose que chaque échantillon de taille k est équiprobable. Une arête de G relie deux sommets d'un échantillon de taille k avec la probabilité

$$\frac{\binom{n-2}{k-2}}{\binom{n}{k}} = \frac{k(k-1)}{n(n-1)}$$

puisqu'il y a $\binom{n-2}{k-2}$ échantillons contenant les extrémités de l'arête. Un sous-ensemble de k sommets n'est pas indépendant si et seulement si au moins une des m arêtes de G relie deux de ses sommets. La probabilité de cet événement est majorée par $m\frac{k(k-1)}{n(n-1)}$. Par conséquent un échantillon est indépendant avec probabilité au moins $1 - m\frac{k(k-1)}{n(n-1)}$. Si cette valeur est positive il y a nécessairement (au moins) un sous-ensemble de k sommets indépendants, ce qui est le cas si $k = \lfloor n/\sqrt{m} \rfloor$. □

On peut obtenir une autre minoration :

Théorème 1.31 (Túran) *Tout graphe G à n sommets et m arêtes contient un sous-ensemble de sommets indépendants de taille α_G au moins égale à $n^2/(2m+n)$.*

Preuve : On numérote les sommets de 1 à n et on associe à toute permutation π de $[1,n]$ l'ensemble, E_π, des sommets u de G tels que $\pi(u) > \pi(v)$ pour tous les voisins v de u. On voit que E_π est un ensemble de sommets indépendants dans G. L'espérance de la taille de E_π est donc un minorant pour α_G. La probabilité que le sommet i soit dans E_π est la probabilité que $\pi(i) = \max\{\pi(j) \,|\, j = i \text{ ou } j \text{ est voisin de } i\}$. Si d_i est le degré de i dans G alors, en posant toutes les permutations équiprobables, la probabilité de cet événement est $1/(d_i+1)$. Par conséquent

$$E(|E_\pi|) = \sum_{i=1}^{n} \frac{1}{d_i+1}$$

Le théorème se déduit de la relation sommets/arêtes et du fait que la moyenne arithmétique majore la moyenne harmonique. □

1.8.4 Arbre binaire de recherche aléatoire

Un arbre binaire de recherche aléatoire est obtenu en insérant les valeurs successives d'une permutation aléatoire dans un arbre binaire de recherche vide au départ. On s'intéresse à la hauteur moyenne d'un arbre de recherche aléatoire

On considère la variable aléatoire $Y_n^{(i)}$ valant la profondeur de la clé de rang i (i.e. de i si on prend $[1,n]$ pour l'ensemble des clés) dans un arbre de recherche aléatoire sur n clés. Si Y_n est la variable aléatoire valant la hauteur d'un arbre de recherche aléatoire alors $Y_n = \max\{Y_n^{(1)}, \ldots, Y_n^{(n)}\}$.

En utilisant l'inégalité de Jensen on peut écrire

$$E(Y_n) \leq \log E(2^{Y_n}) = \log E(2^{\max\{Y_n^{(1)},\ldots,Y_n^{(n)}\}}) < \log E(\sum_{\text{i est une feuille}} 2^{Y_n^{(i)}}).$$

On pose $Z_n = \sum_{\text{i est une feuille}} 2^{Y_n^{(i)}}$, alors

$$E(Z_n) = \frac{1}{n} \sum_{1\leq i\leq n} E(Z_n|\text{racine a rang i}) = \frac{2}{n} \sum_{1\leq i\leq n} (E(Z_{i-1}) + E(Z_{n-i})).$$

En posant $z_n = E(Z_n)$, on a $z_0 = 0, z_1 = 1$ et

$$z_n = \frac{4}{n} \sum_{1\leq i\leq n-1} z_i.$$

Par conséquent on a pour $n \geq 3, nz_n - (n-1)z_{n-1} = 4z_{n-1}$. D'où

$$\frac{z_n}{(n+3)(n+2)(n+1)} = \frac{z_{n-1}}{(n+2)(n+1)n} = \ldots = \frac{1}{30}$$

On en déduit $z_n = O(n^3)$ puis $E(Y_n) = O(\log n)$.

Théorème 1.32 *La hauteur moyenne d'un arbre de recherche aléatoire sur n clés est un $O(\log n)$.*

1.8.5 Echantillonnage aléatoire

Étant donné un ensemble E à n éléments, un r-échantillon aléatoire de E est un sous-(multi-)ensemble à r éléments de E tiré selon une certaine distribution. Les distributions les plus courantes sont les suivantes.

- **Loi uniforme :** On munit l'ensemble des sous-ensembles de taille r de E, noté $\binom{E}{r}$ de la loi uniforme. Chaque r-échantillon a donc la même probabilité d'occurence.
- **Tirage avec remise :** On munit E de la loi uniforme. Un r-échantillon est alors un échantillon de E^r munit de la loi produit. Dit autrement on obtient un r-échantillon en tirant au hasard de manière indépendante r fois un élément de E. Un même élément peut donc se retrouver plusieurs fois dans un r-échantillon.
- **Échantillonage de Bernoulli :** Chaque élément de E est tiré indépendamment avec une (loi de Bernoulli de) probabilité r/n. Un r-échantillon a donc r éléments en moyenne mais peut posséder plus ou moins d'éléments.

Exercice 1.33 *Expliciter pour chacune des trois lois précédentes la probabilité d'occurrence d'un échantillon donné. Vérifier que la somme des probabilités sur l'ensemble des "échantillons" vaut 1.*

Ces trois distributions donnent des résultats asymptotiques semblables dans la pratique pour les calculs de complexité en moyenne. L'utilisation de l'une ou l'autre des distributions se justifie souvent par la commodité des calculs (voir Mulmuley [Mul94, chap. 5]).

1.9 Master theorem

Théorème 1.34 *Soit la relation de récurrence sur $\mathbb{N}^*$*

$$T(n) = aT(n/b) + f(n)$$

avec $a \geq 1$, $b > 1$. Ici $T(n/b)$ peut désigner $T(\lfloor n/b \rfloor)$ ou $T(\lceil n/b \rceil)$. Alors

$$T(n) = \begin{cases} \Theta(n^{\log_b a}) & \text{si } f(n) = O(n^{\log_b a - \epsilon}) \text{ pour un certain } \epsilon > 0 \\ \Theta(f(n)) & \text{si } f(n) = \Omega(n^{\log_b a + \epsilon}) \text{ et } af(n/b) \leq cf(n) \\ & \text{pour } c < 1 \text{ et pour } n \text{ assez grand} \\ \Theta(n^{\log_b a} \log n) & \text{si } f(n) = \Theta(n^{\log_b a}). \end{cases} \tag{1.1}$$

Preuve : Voir par exemple [CLRS02, sec. 4.3] pour une preuve pas à pas. Sinon, en posant $U(n) = T(b^n)/a^n$ on obtient $U(n) = U(n-1) + f(b^n)/a^n$. D'où $U(n) = U(0) + \sum_{1 \leq i \leq n} f(b^i)/a^i$. Notons que $U(0) = T(1)$. On obtient assez simplement

$$\sum_{1 \leq i \leq n} f(b^i)/a^i = \begin{cases} \Theta(1) & \text{si } f(n) = O(n^{\log_b a - \epsilon}) \text{ pour un } \epsilon > 0 \\ \Theta(f(b^n)/a^n) & \text{si } af(n/b) \leq cf(n) \text{ pour } c < 1 \text{ et } n \text{ assez grand} \\ \Theta(n) & \text{si } f(n) = \Theta(n^{\log_b a}). \end{cases}$$

On en déduit $U(n) = O(1)$ dans le premier cas, $U(n) = \Theta(f(b^n)/a^n)$ dans le second cas compte tenu de $f(n) = \Omega(n^{\log_b a+\epsilon})$, et $U(n) = \Theta(n)$ dans le dernier cas. □