Géométrie Algorithmique
Notes de Cours

Francis Lazarus

Octobre 2008

Table des matières

Chapitre 1

Introduction et Rappels

1.1 Qu'est-ce que la géométrie algorithmique ?

La géométrie algorithmique est une discipline récente (circa 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 :

- 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`

- Computational Geometry. Algorithms and applications. de Berg, van Kreveld, Overmars and Schwarzkopf. Springer 1997. (Une bonne introduction)

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

- 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)

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

- 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

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. 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 seul 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 programme exécute au plus $f(n)$ instructions. Il s'agit donc de la complexité dans le cas le pire relativement à tous les jeux de données de taille n.

On peut également définir la taille de l'entrée par le nombre total de bits pour la coder, c'est à dire la somme du nombre de bits dans chaque registre d'entrée. Dans ce cas, 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 $+, -, *, /, <, =$) 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. *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.

- 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 de coût unitaire 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($\square$) = 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 $\mathrm{lce(b)} \geq \mathrm{lce}(b_t) = (t+1)\log(t+1)$ aux ordres convenables on a

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

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 1.15).*

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.

Lemme 1.8 *L'espérance est linéaire.*

Exercice : 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 1.9 *Si X est à valeurs naturelles alors*

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

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

Lemme 1.10 *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 1.11 (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 1.12 (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 1.13 (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 1.14 *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 1.15 *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 1.16 *Soient deux événements $B \subset A$ avec $P(B) > 0$ et X une variable aléatoire. Montrer que $(X|A)|B = X|B$.*

Lemme 1.17 *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 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 le constante ou borne dans la seconde égalité et d'utiliser le fait que $\sum_{i\in I} P(B_i) = 1$.

Exercice 1.18 *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 1.19 *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 1.20 *Soient A un événement et X une variables aléatoire. Montrer une version conditionnelle du lemme 1.17, 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 1.21 *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 1.22 *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 1.23 (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 1.24 (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 1.25 (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 1.26 (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 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 1.27 *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*

$$\forall x > 0 : \qquad P(X \geq x) < \exp(-x^2/(2n))$$

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 1.28 *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 1.17).
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 1.29 *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 1.30 (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 1.31 *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 1.32 *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

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

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

avec $a \geq 1$, $b > 1$ alors

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

Chapitre 2

Graphes planaires

Les graphes planaires apparaissent sous la forme de réseaux routiers, de frontières dans les cartes géographiques, de circuits imprimés ou encore dans un cadre plus théorique, comme les graphes sommets/arêtes (1-squelettes) des 3-polytopes. La plupart des propriétés des graphes planaires font appel de manière plus ou moins explicite au fameux théorème de Jordan.

Sauf avis contraire, on considère dans ce qui suit des graphes *simples*, i.e. sans arête multiple ni boucle.

2.1 Plongements

Définition 2.1 *Un* plongement *d'un graphe $G = (S, A)$ dans un espace X est la donnée d'une injection $S \hookrightarrow X$ et pour chaque arête $a \in A$, d'un plongement $p_a : [0, 1] \to X$ dont les extrémités coïncident avec l'injection de celles de a, de sorte que les plongements de deux arêtes ne s'intersectent qu'en leurs extrémités communes, le cas échéant. On appelle* arc *le plongement d'une arête.*

Définition 2.2 *Un graphe est* planaire *s'il peut être plongé dans le plan. Un graphe* plan *est un graphe plongé dans le plan. Dit autrement un graphe plan est un plongement particulier d'un graphe planaire. Un plongement d'un graphe dont tous les arcs sont polygonaux est dit* polygonal - ou PL (pour piecewise linear). *Une* face *d'un graphe plan est une composante connexe du complémentaire du plongement du graphe dans le plan.*

Par la suite on utilisera la même notation pour un graphe plan et son plongement. Ainsi, si s est un sommet du graphe plan G, alors $G - s$ désignera soit le graphe G privé de s et des arêtes incidentes à s, soit son plongement, i.e. le plongement de G privé de s et des *arcs* correspondant aux arêtes incidentes à s.

Lemme 2.3 *Un ouvert de $\mathbb{R}^2$ connexe (par arcs) est connexe par arcs polygonaux simples.*

Preuve : Soit $\gamma : p \rightsquigarrow q$ un chemin dans un ouvert Ω connexe. Considérer le sup des $t \in [0,1]$ tels qu'il existe un chemin polygonal simple reliant x à $\gamma(t)$ dans Ω. Montrer que ce nombre vaut nécessairement 1. □

Lemme 2.4 *Tout graphe planaire admet un plongement polygonal.*

Preuve : Considérons un plongement d'un graphe planaire. On choisit pour chaque sommet p du plongement un disque D_p de centre p de sorte que D_p n'intersecte que les arcs incidents à p et que deux tels disques D_p et D_q soient disjoints. Pour chaque arc joignant p à q, on considère une composante $C_{p,q}$ de cet arc joignant D_p à D_q. Par le lemme précédent, $C_{p,q}$ peut être remplacé par un arc polygonal $C'_{p,q}$ dans le plan privé des autres disques et des autres arcs. On considère un sous arc $C"_{p,q}$ de $C'_{p,q}$ joignant D_p à D_q sans rencontrer l'intérieur de ces disques. Finalement, en prolongeant tous les $C"_{p,q}$ arcs ainsi obtenus par des segments de droites joignant les centres des disques D_p et D_q, on obtient un plongement polygonal. □

2.2 Le théorème de Jordan

Le théorème de Jordan (du mathématicien français Camille Jordan. 1838 - 1922) établit qu'une courbe fermée simple du plan sépare le plan en deux composantes bordées par cette courbe. Ce résultat évident en apparence est singulièrement difficile à montrer. La preuve qui suit est tirée de
A Proof of the Jordan Curve Theorem. Helge Tverberg. Bull. London Math. Soc. 12(1980), pp. 34-38.
Elle consiste à montrer ce théorème pour les courbes polygonales et à l'attendre aux courbes continues par un processus de passage à la limite.

Une autre preuve a été donnée par Thomassen dans *The Jordan-Schönflies Theorem and the classification of surfaces.* Carsten Thomassen. American Mathematical Monthly. Feb 1992. pp 116-129.
également reprise dans
Graphs on Surfaces. Bojan Mohar et Carsten Thomassen. Johns Hopkins university Press, 2001.
Comme chez Tverberg, Thomassen commence par traiter le cas des courbes polygonales. Le cas général est ramené (de manière élégante en ce qui concerne la non-connexité du complémentaire d'une courbe) à la non-planarité de $K_{3,3}$.

Ces preuves font appel à un minimum de topologie et se limitent plus ou moins aux implications classiques de la compacité pour les applications continues. Les preuves sont de ce fait relativement accessibles bien qu'assez fastidieuses. On trouvera cependant dans les livres de topologie algébriques des preuves plus courtes et plus générales (où l'on traite des injections d'une (n-1)-sphère dans une n-sphère) faisant appel aux suites de Mayer-Vietoris pour le calcul de l'homologie des espaces en jeu. Voir par exemple
Elements of Algebraic Topology. James Munkres. Perseus Books, 1984.

Le cas plus restreint des courbes différentiables est traité dans

Géométrie différentielle : variétés, courbes et surfaces. Marcel Berger et Bernard Gostiaux. PUF mathématiques, 1987.

Théorème 2.5 (Jordan - version polygonale) *Soit C une courbe polygonale fermée simple. Alors $\mathbb{R}^2 \setminus C$ a deux composantes, l'une bornée, l'autre non-bornée, toutes deux bordées par C.*

Preuve : Notons tout d'abord que C étant compact $\mathbb{R}^2 \setminus C$ a exactement une composante non bornée. Soit une direction $\vec{d}$ transverse aux segments de C que l'on appellera direction horizontale. On considère les segments de C semi-ouverts supérieurement (s.o.s), i.e. privés de leur sommet supérieur. On note $\pi(z)$ la parité du nombre de segments s.o.s de C coupés par la demi-droite horizontale $(z, \vec{d})$. On vérifie que π est localement constante dans $\mathbb{R}^2 \setminus C$ (regarder les arêtes qui coupent une petite bande horizontale centrée autour de z). Donc π est constante sur chaque composante de $\mathbb{R}^2 \setminus C$. De plus π prend des valeurs distinctes de chaque côté d'un segment de C. Il suit que $\mathbb{R}^2 \setminus C$ a au moins deux composantes. Soit V_C un voisinage tubulaire de C, et soit D un petit disque intersectant C en un segment. Considérons alors p, q, r trois points de $\mathbb{R}^2 \setminus C$. Il existe trois chemins dans $\mathbb{R}^2 \setminus C$ joignant p, q, r à V_C (exemple : des segments de droite). En prolongeant ces chemins dans $\mathbb{R}^2 \setminus C$ on peut s'arranger pour qu'ils joignent D en restant dans $\mathbb{R}^2 \setminus C$. Comme $D \cap (\mathbb{R}^2 \setminus C)$ n'a que deux composantes, deux des trois points p, q, r sont dans la même composante de $\mathbb{R}^2 \setminus C$. On en déduit que $\mathbb{R}^2 \setminus C$ a exactement deux composantes. Par ailleurs les arguments qui précédent montrent également que tout point de C est adhérent aux deux composantes de $\mathbb{R}^2 \setminus C$. □

Avant de passer à la version générale du théorème, voici quelques applications.

Corollaire 2.6 (Lemme du θ) *Soient C_1, C_2, C_3 trois courbes polygonales simples (non fermées) ayant en commun leurs extrémités p et q. Alors le graphe $G = C_1 \cup C_2 \cup C_3$ a précisément 3 faces respectivement bordées par $C_1 \cup C_2$, $C_2 \cup C_3$ et $C_3 \cup C_1$.*

Preuve : Par la version polygonale du théorème de Jordan, les trois courbes fermées simples $G_k = C_i \cup C_j$, $\{i, j, k\} = \{1, 2, 3\}$, séparent le plan en deux composantes et bordent ces mêmes composantes. On note X_k (resp. Y_k) la face bornée (resp. non-bornée) de G_k. On note également $\overset{\circ}{C}_i = C_i \setminus \{p, q\}$ l'intérieur relatif de C_i.

Remarquons qu'une courbe polygonale simple (ici privée de ses extrémités) ne peut séparer un ouvert connexe en plus de deux composantes (cf. la preuve du théorème de Jordan polygonal). Comme $\overset{\circ}{C}_3$ est inclus dans l'une des faces de G_3 on en déduit que $G = G_3 \cup \overset{\circ}{C}_3$ a au plus trois faces.

Par ailleurs on a $\overset{\circ}{C}_i \subset X_i$ pour au moins un indice $i \in \{1, 2, 3\}$. Dans le cas contraire on a $C_i \subset \complement X_i$ et donc $G_i \subset G \subset \complement X_i$, d'où $X_i \subset \complement G \subset \complement G_i$. Dit autrement X_i est une face de G. Comme les X_i sont distincts ($C_i \subset \bar{X}_j$ mais $\overset{\circ}{C}_i \not\subset \bar{X}_i$) on en conclut que G a au moins trois faces bornées et donc au moins quatre faces ce qui contredit la remarque précédente. On supposera par la suite $\overset{\circ}{C}_3 \subset X_3$.

De $G = G_1 \cup G_2$ on tire que toute face de G est une composante de l'intersection d'une face de G_1 avec une face de G_2. De $G_3 \subset G \subset \complement Y_3$ on tire que Y_3 est une face de G. Comme Y_3 est non bornée on a $Y_3 \subset Y_1 \cap Y_2$.

Comme $C_1 \subset \bar{Y}_3 \subset \bar{Y}_1 = \complement X_1$, on a en fait $G_1 \subset G \subset \complement X_1$ et donc X_1 est une face de G. De même X_2 est une face de G. Or, Y_3 qui est non bornée est distincte des deux faces bornées X_1 et X_2, elles mêmes distinctes (C_1 borde X_2 mais pas X_1). On conclut que Y_3, X_1 et X_2 sont les trois faces de G. □

Définition 2.7 *Un graphe G est* 2-connexe *s'il a trois sommets au moins et si pour chacun de ses sommets, s, le graphe $G - s$ est connexe.*

Proposition 2.8 *Toute face d'un graphe plan polygonal 2-connexe est bordée par un cycle de ce graphe. De plus tout arc du graphe est incident (i.e. adhérent) a exactement deux faces.*

Preuve : Soit G un tel graphe. On raisonne par récurrence sur $\sum_{s \in S(G)} (d(s) - 2)$. Si cette somme est nulle alors G est un cycle et on peut appliquer le théorème de Jordan. Sinon, on considère une chaîne maximale P dans G dont les sommets intérieurs sont de degré 2. On vérifie que $G - P$ est 2-connexe et on peut lui appliquer l'hypothèse de récurrence, puisque la somme ci-dessus diminue. Il suit que P est contenue dans une face de $G - P$ qui est bordée par un cycle de $G - P$. On applique alors le lemme du θ à l'union de ce cycle et de P pour conclure. □

Lemme 2.9 *Soit G un graphe plan polygonal, s un sommet de degré un de G et a l'arête incidente à s. Alors G a le même nombre de faces que $G - s$.*

Preuve : Comme $G - s$ est inclus dans G, toute face de G est incluse dans une face de $G - s$. Soient f_1 et f_2 deux faces de G incluses dans une même face de $G - s$ et soient p_1 et p_2 deux points respectivement intérieurs à f_1 et f_2. Alors il existe un chemin polygonal P joignant p_1 et p_2 dont l'intersection avec G est incluse dans l'arc semi-ouvert $a \cup s$. En considérant un petit voisinage tubulaire de $a \cup s$, on peut modifier P en contournant a de manière à éviter G, ce qui montre que $f_1 = f_2$. □

Théorème 2.10 (Relation d'Euler) *Les nombres F, A et S de faces, arêtes et sommets d'un graphe plan polygonal connexe vérifient la relation, dite d'Euler (1707-1783),*

$$F - A + S = 2$$

Preuve : Par récurrence sur A. Si A est nul (et donc $S = 1$) la relation est trivialement vraie. Soit G un graphe avec $A > 0$ arêtes. Si G contient un sommet s de degré un, alors par le lemme 2.9, $G - s$ a F faces et l'hypothèse de récurrence appliquée à $G - s$ permet de confirmer la relation d'Euler pour G. Sinon G contient un cycle simple C. Soit a une arête de C. Montrons que $G - a$ a une face de moins que G ce qui permettra de conclure avec l'hypothèse de récurrence. D'après la version polygonale du théorème de Jordan, C

sépare le plan en deux régions bordées par C. Comme G est la réunion de C et $G-a$, toute face de G est incluse dans l'intersection d'une face de C et d'une face de $G-a$. L'arc a est inclus dans une unique face de $G-a$, disons f. Toute face de $G-a$ distincte de f ne rencontre pas C et est donc une face de G. Comme f intersecte les deux faces de C, on en déduit que G a au moins une face de plus que $G-a$. Mais en considérant un voisinage tubulaire de a dans f, on montre par un raisonnement déjà vu que $f-a$ a au plus deux composantes. Donc G a au plus une face de plus que $G-a$. Finalement G a exactement une face de plus que $G-a$. □

Pour une série de preuves plus ou moins formelles de cette fameuse formule on pourra consulter la page :
http ://www.ics.uci.edu/∼eppstein/junkyard/euler/
extraite du par ailleurs très intéressant site : "The Geometry Junkyard" maintenu par David Eppstein.

Théorème 2.11 *Le graphe complet K_5 et le graphe bipartite complet $K_{3,3}$ ne sont pas planaires.*

Preuve : Supposons $K_{3,3}$ planaire. Par le lemme 2.4, on peut lui appliquer la relation d'Euler, ce qui fournit $F = 5$. Par ailleurs toute face d'un plongement de $K_{3,3}$ est bordée par un cycle (cf. proposition 2.8) de $K_{3,3}$ ayant au moins 4 arêtes (tout cycle d'un graphe bipartite est de longueur paire !). Par la relation d'incidence face/arête (cf. proposition 2.8) on en déduit $4F \leq 2A$. Une contradiction.

Le cas de K_5 se traite de manière similaire. □

Passons maintenant au théorème de Jordan dans sa version générale (plane).

Théorème 2.12 (Jordan) *Soit C une courbe fermée simple, i.e. l'image injective par une application continue du cercle unité S^1 dans le plan. Alors $\mathbb{R}^2 \backslash C$ a deux composantes, l'une bornée, l'autre non-bornée, toutes deux bordées par C.*

Preuve : Je donne sans entrer dans tous les détails la preuve de Tverberg (1980).

Tout d'abord C peut être approximée d'aussi près que l'on veut par une courbe polygonale. Dit autrement, pour tout ϵ positif on peut trouver un polygone de Jordan (= une courbe polygonale simple fermée) C' tel que

$$|C - C'| = \sup_{x \in S^1} |C(x) - C'(x)| < \epsilon.$$

La preuve consiste à couvrir le plan d'une grille à mailles carrées de côté δ (défini plus loin). On pose $C_0 = C$ et on considère un à un les carrés de la grille traversés par C (ils sont en nombre fini). Pour chaque tel carré M_i, on remplace l'image par C_i du plus petit arc de S^1 contenant $C_i^{-1}(M_i)$ par le segment de droite joignant ses extrémités, obtenant ainsi C_{i+1}. C_n, où n est le nombre de carrés traversés, est la courbe C' cherchée. Pour définir δ on choisit tout d'abord ϵ_1 tel que

$$|x - y| < \epsilon_1 \implies |C(x) - C(y)| < \epsilon/2,$$

puis ϵ_2 tel que

$$|C(x) - C(y)| < \epsilon_2 \implies |x - y| < \min(\epsilon_1, \sqrt{3}).$$

L'existence de ϵ_1 découle de l'uniforme continuité de C sur le compact S^1. Celle de ϵ_2 de l'uniforme continuité de C^{-1}, inverse d'une bijection continue sur le compact $C(S^1)$. On pose alors $\delta = \min(\epsilon/2, \epsilon_2)$. Il reste à vérifier que C_n ainsi construit convient. (La solution est dans l'article de Tverberg).

Un premier lemme : Si C' est un polygone de Jordan, alors la composante bornée de $\mathbb{R}^2 \backslash C'$ contient un disque touchant C' en deux points $C'(x)$ et $C'(y)$ tels que $|x - y| > \sqrt{3}$.

Pour la preuve, on considère un disque D comme dans le lemme avec $|x-y|$ maximal. Si $|x - y| < \sqrt{3}$ alors tout point z sur le plus grand arc A de S^1 joignant x et y est à une distance de x ou de y supérieure à celle de x à y. On en déduit par l'hypothèse sur x et y que, en dehors de ses extrémités, $C'(A)$ ne touche pas le bord de D. Une construction simple, dépendant du fait que D est tangent ou non en $C'(x)$ et $C'(y)$, permet alors de remplacer D par un disque D' touchant C' en $C'(x')$ et $C'(y')$ tels que $|x' - y'| > |x - y|$, contredisant ainsi l'hypothèse sur x et y. (voir les détails dans l'article de Tverberg).

Un second lemme : Soit C' un polygone de Jordan, et deux points a et b dans la même composante de $\mathbb{R}^2 \setminus C'$. Si la distance de a et de b à C' est supérieure ou égale à 1, et si aucune corde de C' de longueur inférieure à 2, ne sépare a de b, alors il existe un chemin continue Π de a à b tel que $d(\Pi, C') \geq 1$.

On commence par remarquer que l'implication du lemme est en fait une équivalence. Il suit que si a et b sont reliés à a' et b' par des chemins distants d'au moins 1 de C', alors les hypothèses sur a et b sont valides pour a' et b'. On peut donc supposer que a et b sont à distance exactement 1 de C'. On considère le disque de rayon 1 centré en a et l'idée est de définir Π comme le lieu du centre de ce disque roulant le long de C' jusqu'à atteindre b. Comme ce disque doit rester intérieur à C', il devra possiblement court-circuiter des arcs de C' et il s'agit de montrer qu'on peut malgré tout atteindre b dans tous les cas. On utilisera pour cela l'hypothèse sur les cordes. (voir les détails dans l'article de Tverberg).

Terminons par la preuve du théorème.

Premièrement, $\mathbb{R}^2 \setminus C$ a au moins deux composantes :

Montrons qu'en plus d'une composante non bornée, il en existe une bornée. On considère pour cela un disque D_0 contenant C, et une suite $(C_n)_{n>0}$ de polygones de Jordan contenus dans D_0 et convergeant vers C. Par le premier lemme, chaque C_n contient un disque D_n touchant C_n en deux points $C_n(x_n)$ et $C_n(y_n)$ tels que $|x_n - y_n| > \sqrt{3}$. Soit z_n le centre de D_n, et soit z la limite d'une sous-suite convergente de (z_n). On va montrer que z ne peut être dans la composante non bornée de $\mathbb{R}^2 \setminus C$ en se ramenant au cas des polygones de Jordan, déjà traité. Par compacité, $|C(x_n) - C(y_n)|$ est borné inférieurement. Par convergence, il en est de même de $|C_n(x_n) - C_n(y_n)|$. Ce qui implique la même chose pour le rayon de D_n et donc pour $d(z_n, C_n)$. Comme $z_n \to z$, z est dans D_n pour n assez grand et donc intérieur à C_n. Si z était hors de D_0, alors on pourrait trouver un chemin Π le reliant à un point hors de D_0. Par compacité, $d(\Pi, C) > 0$, donc $d(\Pi, C_n) > 0$ pour n assez grand. Donc z est hors de C_n, une contradiction.

Deuxièmement, $\mathbb{R}^2 \setminus C$ a au plus deux composantes :

Soient p, q et r trois points de $\mathbb{R}^2 \setminus C$. On veut montrer que deux de ces points sont nécessairement dans une même composante. On considère à nouveau une suite $(C_n)_{n>0}$ de polygones de Jordan convergeant vers C. Pour n assez grand p, q et r sont dans $\mathbb{R}^2 \setminus C_n$ (i.e. à une distance non nulle de C_n). Quitte à prendre une sous-suite on peut supposer que p et q sont dans une même composante de $\mathbb{R}^2 \setminus C_n$. Si pour une infinité de n, p peut être relié à q dans $\mathbb{R}^2 \setminus C_n$ par un chemin Π_n tel que $d(\Pi_n, C_n)$ est borné inférieurement, alors il en sera de même pour $d(\Pi_n, C)$ à partir d'un certain rang, ce qui montre que p et q sont dans une même composante de $\mathbb{R}^2 \setminus C$. Sinon, par le second lemme on peut trouver une suite de segments $[C_n(x_n), C_n(y_n)]$ séparant p et q dans $\mathbb{R}^2 \setminus C_n$, et dont la longueur tend vers 0. On en déduit que $|C(x_n) - C(y_n)|$ tend vers 0 et donc que $|x_n - y_n|$ tend vers 0, par continuité uniforme de C^{-1}. Mais alors la plus petite des deux composantes de $\mathbb{R}^2 \setminus C_n \cup [C_n(x_n), C_n(y_n)]$ converge vers un point, impliquant que p ou q est sur C_n, une contradiction. □

On a coutume d'appeler une courber fermée simple du plan, un *courbe de Jordan*. Notons qu'en modifiant le deuxièmement de la preuve, on montre sans trop de difficultés que

Théorème 2.13 *Un arc simple du plan (i.e. l'image continue injective du segment* $[0, 1]$*) ne sépare pas le plan.*

Une version plus forte du théorème de Jordan stipule que l'intérieur d'une courbe de Jordan est un disque topologique :

Théorème 2.14 (de Jordan-Schönflies) *Tout homéomorphisme entre deux courbes de Jordan s'étend au plan tout entier.*

Une preuve est donnée dans
The Jordan-Schönflies Theorem and the classification of surfaces. Carsten Thomassen. American Mathematical Monthly. Feb 1992. pp 116-129.

Sans entrer dans les détails, Thomassen commence par considérer une famille dénombrable de points qui soit dense dans la première courbe, disons C, et qui puissent être joint à tout point de l'intérieur de C par un chemin polygonal. Il considère également une famille dénombrable de points qui soit dense dans l'intérieur de C. On peut alors en déduire une suite où chaque point de ces deux familles apparaît une infinité de fois. On construit alors récursivement un homéomorphisme entre l'union de C et des n premiers points de la suite (plus d'autres points autour) et l'union de la seconde courbe, que l'on peut supposer polygonale, et de n points (plus d'autres points autour) qui vont également remplir l'intérieur de cette courbe polygonale. On obtient à la limite une application définie sur C et la suite de points. On montre que cette application se prolonge sur l'adhérence de l'intérieur de C en un homéomorphisme. Pour définir un homéomorphisme sur le plan tout entier, on commence par entourer les deux courbes d'un grand carré, T, que l'on relie aux deux courbes, ce qui permet d'étendre l'homéomorphisme sur ces courbes, en prenant l'identité sur T. La méthode précédente permet d'étendre cet homéomorphisme sur l'intérieur de T que l'on prolonge finalement par l'identité en dehors de T.

À l'aide de ce théorème on montre un analogue général au lemme du thêta et à la relation d'Euler.

2.3 Graphes interdits

Un des résultats les plus célèbres sur la planarité est la réciproque du théorème 2.11 due au mathématicien Kazimierz Kuratowski (1896 - 1980), stipulant qu'un graphe ne "contenant" ni K_5 ni $K_{3,3}$ est planaire. Les graphes K_5 et $K_{3,3}$ sont ainsi appelés *graphes interdits* ou graphes de Kuratowski. De manière plus générale on montre que pour chaque surface de genre g il existe un nombre fini de graphes interdits empêchant un graphe quelconque d'être plongé sur cette surface.

Théorème 2.15 (de Kuratowski) *Un graphe est planaire si et seulement si il ne contient pas de subdivision de K_5 ou de $K_{3,3}$ comme sous-graphe.*

La preuve qui suit, tirée de
Graphs on Surfaces. Bojan Mohar et Carsten Thomassen. Johns Hopkins university Press, 2001.
repose sur trois lemmes, par ailleurs intéressants en eux-mêmes.

Lemme 2.16 *Soit G un graphe 3-connexe ayant au moins 5 sommets. Alors G admet une arête e telle que $G//e$ (i.e. la suppression de e, suivie de l'identification des extrémités de e, suivie de la fusion des éventuelles arêtes multiples en arêtes simples de mêmes extrémités) est 3-connexe.*

Preuve : Supposons par l'absurde que pour toute arête $e = xy$ de G, $G//e$ n'est pas 3-connexe. Alors il existe $z, t \in S(G//e)$ (les sommets de $G//e$) qui déconnectent $G//e$, où t résulte nécessairement de l'identification de x et y. Dit autrement, pour toute arête $e = xy$, il existe $z \in S(G)$ tel que $G - \{x, y, z\}$ n'est pas connexe. On choisit e et z de sorte que la plus grande (en nombre de sommets) des composantes de $G - \{x, y, z\}$ soit maximale. Soit H cette composante et soit u adjacent à z dans une autre composante de $G - \{x, y, z\}$, H'. D'après ce qui précède, il existe $v \in S(G)$ tel que $G - \{z, u, v\}$ ne soit pas connexe. Montrons que le sous-graphe H'' de G induit par $(S(H) \cup \{x, y\}) \setminus \{v\}$ est dans une composante de $G - \{z, u, v\}$. Comme ce sous-graphe a plus de sommets que H, on aboutit à une contradiction. H'' est connexe car tout sommet t de H peut être relié à x ou y (eux même reliés par l'arête e) dans H'' : par la 3-connexité de G, on a l'existence d'un chemin $p : t \rightsquigarrow x$ dans G qui évite z et v. On écrit

$$p = (p_1, w, p_2) \text{ où } w \in \{x, y\} \text{ et } x \notin p_1, y \notin p_1.$$

Alors le chemin p_1 est dans une composante de $G - \{x, y, z\}$ donc dans H. Et comme il ne contient pas v, on en déduit que (p_1, w) est dans H''. Par ailleurs H'' est dans $G - \{z, u, v\}$; il est donc dans l'une des composantes de $G - \{z, u, v\}$. □

Lemme 2.17 *Soit G un graphe 3-connexe ne contenant pas de subdivision de K_5 ou de $K_{3,3}$ comme sous-graphe. Alors G admet un plongement* rectiligne convexe *(i.e. dont les arêtes sont des segments de droites et dont les faces sont des convexes) dans le plan.*

Preuve : Notons que ce lemme implique une version du théorème de Kuratowski restreinte aux graphes 3-connexes. Pour la preuve, on raisonne par récurrence sur le nombre de sommets de G. Le résultat se vérifie à la main si ce nombre vaut 4 ou 5. Par le précédent lemme on peut choisir une arête $e = xy$ telle que $G' = G//e$ est 3-connexe. Clairement G' ne contient pas de subdivision d'un graphe interdit (on vérifie sinon que se serait le cas pour G). Par l'hypothèse de récurrence G' possède un plongement rectiligne convexe. Soit z le sommet de G' résultant de l'identification de x et y. $G' - z$ est 2-connexe. Par la proposition 2.8, on considère le cycle C de $G' - z$ bordant la face de $G' - z$ contenant z. Soit X (resp. Y) l'ensemble des extrémités des arêtes de G reliant x (resp. y) à ce cycle. On vérifie que X et Y ne peuvent se chevaucher (i.e. $|X \cap Y| < 3$ et il n'y a pas deux sommets de X et deux sommets de Y apparaissant de manière alternée autour de C). Dans le cas contraire on met en évidence un graphe interdit dans G. Ceci permet de construire un plongement convexe pour G en remplaçant z par x dans le plongement de G' et en insérant y dans la face bordée par x et par le segment de C où se rattache les arêtes incidentes à y. Notons que si z est sur la face externe de G', une transformation projective permet de rendre cette face interne. □

Le lemme suivant permet de se ramener au précédent dans tous les cas :

Lemme 2.18 *Soit G un graphe ne contenant pas de subdivision de K_5 ou de $K_{3,3}$ comme sous-graphe et tel que l'ajout d'une arête entre deux sommets non-adjacents crée un tel sous-graphe. Alors G est 3-connexe.*

Preuve : On raisonne par récurrence sur le nombre de sommets de G. Le résultat se vérifie à la main si ce nombre vaut 4 ou 5.

G est 2-connexe :
sinon on peut écrire $G = G_1 \cup G_2$ où G_1 et G_2 n'ont qu'un sommet x en commun. Soit $y_i \in G_i, i = 1, 2$, adjacent à x. L'ajout d'une arête y_1y_2 à G crée par hypothèse une subdivision K d'un graphe interdit. Comme ceux-ci sont trois connexes et qu'il n'y a que les 2 passages x et y_1y_2 entre G_1 et G_2, les sommets de degré 3 de K sont tous dans G_1 ou tous dans G_2. Mais on peut alors remplacer le chemin de K comportant l'arête y_1y_2 et le sommet x par y_1x ou y_2x pour faire apparaître une subdivision d'un graphe interdit dans G. Une contradiction.

Si $G - \{x, y\}$ n'est pas connexe alors xy est une arête de G :
sinon on peut écrire $G = G_1 \cup G_2$ où G_1 et G_2 n'ont que les sommets x et y en commun. L'ajout d'une arête xy à G crée par hypothèse une subdivision K d'un graphe interdit. Comme précédemment les sommets de degré 3 de K sont tous un même G_i, disons G_1. Mais on peut alors remplacer l'arête xy dans K par un chemin reliant x et y dans G_2 (qui contient nécessairement un tel chemin) faisant ainsi apparaître une subdivision d'un graphe interdit dans G. Une contradiction.

Supposons que G n'est pas 3-connexe et soient x, y deux sommets qui déconnectent G. On écrit $G = G_1 \cup G_2$ où G_1 et G_2 n'ont que les sommets x et y et l'arête xy en commun. Il est facile de voir que l'ajout d'une arête à G_i $(i = 1, 2)$ crée une subdivision d'un graphe interdit dans ce même G_i. On peut donc appliquer l'hypothèse de récurrence à G_i et par le lemme précédent choisir un plongement convexe de G_i. Soit alors z_i un autre sommet

du cycle d'une face F_i bordée par x et y dans ce plongement. L'ajout d'une arête z_1z_2 à G crée une subdivision K d'un graphe interdit. Si tous les sommets de degré 3 sont dans un même G_i alors on peut facilement modifier K pour qu'il se trouve dans G_i. Une contradiction. Par ailleurs $S(G_1)\setminus S(G_2)$ ou $S(G_2)\setminus S(G_1)$ ne contient qu'un seul sommet de degré 3 de K. Dans le cas contraire il faudrait au moins 4 chemins disjoints entre G_1 et G_2 dans $G + z_1z_2$. Pour la même raison K ne peut être qu'une subdivision de $K_{3,3}$. Si G_i contient les sommets de degré 3 de K alors on obtient un plongement planaire de $K_{3,3}$ en reliant un point intérieur à F_i aux sommets x, y et z_i. Une contradiction. □

Il existe une autre preuve du lemme 2.17 due à Tutte (1963) et fournissant un algorithme directe de plongement. Tutte montre pour cela que le système linéaire donné par un plongement convexe quelconque d'un cycle facial de G et exprimant les autres sommets de G comme des isobarycentres de leurs voisins admet toujours une unique solution fournissant un plongement convexe.

Notons que le lemme 2.17 et le suivant montrent en particulier que tout graphe planaire admet un plongement rectiligne. Le problème du plongement rectiligne est à l'origine de nombreuses études comme l'existence de plongements dont les sommets sont à coordonnées entières, comme la recherche de la grille entière minimale pouvant contenir un tel plongement, etc...

Par ailleurs il est facile de voir par la relation d'Euler que toute triangulation du plan (dont le cycle externe est également un triangle) est maximale et donc 3-connexe par le lemme précédent.

Enfin, une autre formulation du théorème de Kuratowski affirme qu'un graphe est planaire si et seulement si aucun des 2 graphes interdits n'en est un *mineur*.

On pourra consulter également pour ce qui précède (en particulier pour la preuve – dans le cas polygonal – du théorème de Jordan, de la relation d'Euler et du théorème de Kuratowski) le chapitre 4 du livre de Diestel :

Graph Theory. Reinhard Diestel. Springer-Verlag, Graduate Texts in Mathematics, Volume 173 July 2005 (2000, 1997).

Une copie électronique est disponible à l'adresse :
http ://www.math.uni-hamburg.de/home/diestel/books/graph.theory/GraphTheoryIII.pdf

Voir les articles de TGGT 2008. En particulier par Haeupler and Tarjan.

Chapitre 3

Triangulation

Une triangulation d'une région polygonale du plan est une décomposition de cette région en triangles dont les sommets sont ceux du bord de la région. Une triangulation permet souvent de résoudre plus facilement des problèmes portant sur la région qu'elle triangule. Le problème du gardiennage d'une galerie d'art en est un bel exemple.

Au début du XXe siècle N. J. Lennes montre de manière constructive que tout polygone simple admet une triangulation (Lennes, N., Theorems on the simple finite polygon and polyhedron, Amer. J. Math., 33 (1911), pp. 36-62). Cette construction fournit de fait un algorithme de complexité quadratique en fonction du nombre de sommets. Du temps de l'émergence de la géométrie algorithmique, Garey et al. (1978) ont proposé un algorithme de complexité $O(n \log n)$ pour trianguler un polygone à n côtés. Après diverses améliorations (cf. notes historiques du livre de de Berg et al.), Bernard Chazelle montre en 1991 qu'un polygone simple peut être triangulé en temps linéaire. L'algorithme de Chazelle, décrit dans

- Triangulating a simple polygon in linear time. B. Chazelle. Discrete and Computational Geometry, 6 :485-524, 1991.

est réputé très complexe. Une version plus simple et randomisée est décrite dans

- A randomized algorithm for triangulating a simple polygon in linear time. Amato, Goodrich and Ramos. Discrete and Computational Geometry, 26 :245-265, 2001.

Le problème de la triangulation de l'intérieur d'un polyèdre dans $\mathbb{R}^3$ est beaucoup plus compliqué. Contrairement au cas bidimensionnel le nombre de tétraèdre d'une triangulation d'un polyèdre (même convexe) à n sommets peut varier suivant la triangulation. De plus, tous les polyèdres ne sont pas triangulables, à moins d'ajouter des sommets intérieurs (dits de Steiner), comme le montre le cas du polyèdre de Shönhardt. Ce polyèdre est obtenu à partir d'un prisme de base triangulaire en tournant légèrement le triangle supérieur par rapport au triangle inférieur. Du coup, les faces verticales du prisme (des quadrilatères) ne sont plus planes et il faut ajouter une diagonale pour trianguler chacun de ces quadrilatères gauches. En choisissant cette diagonale de manière à rendre les quadrilatères 'concaves', on vérifie que toute nouvelle arête entre deux sommets du polyèdre est extérieure au polyèdre. Il n'est donc pas possible de trianguler son intérieur.

Références :

- Handbook of Discrete and Computational Geometry. Edited by Goodman and O'Rourke. CRC Press 2004.

3.1 Existence

Définitions La *ligne polygonale* de sommets $(s_1, \ldots, s_n)$ est la suite de segments $(s_1s_2, \ldots, s_{n-1}s_n)$. Cette ligne polygonale est *fermée* si $s_1 = s_n$; elle est simple si deux de ses segments non consécutifs sont disjoints et si deux de ses segments consécutifs s'intersectent en un unique sommet. Un *polygone* est une ligne polygonale simple et fermée. On appelle *arêtes* les segments d'un polygone. Par le théorème de Jordan (version affine), un polygone P sépare le plan en deux régions connexes appelées *intérieur* et *extérieur* de P. Dans ce chapitre, on notera respectivement $IntP$ et $extP$ ces régions (ce sont des ouverts du plan).

Une *diagonale* d'un polygone P est un segment dont l'intérieur relatif (i.e. le segment privé de ses extrémités) est intérieur à P et dont les extrémités sont des sommets de P. Une *triangulation* de P est un recouvrement de son intérieur (au sens large, i.e. de $\overline{IntP}$) par des triangles d'intérieurs disjoints et dont les côtés sont soit des arêtes soit des diagonales de P.

Lemme 3.1 *Tout polygone ayant au moins 4 sommets admet une diagonale.*

Preuve : Soit P un polygone et soit s le sommet de P de coordonnées minimales pour l'ordre lexicographique (s est le sommet le plus bas parmi les sommets les plus à gauche). Soit p le sommet précédant s et q le sommet suivant s pour l'ordre circulaire dans P.
- Si le triangle spq ne contient (au sens large) aucun sommet de $P \setminus \{s, p, q\}$, alors le segment pq est une diagonale : aucune arête de P ne peut rencontrer l'intérieur ni le bord de spq car l'une de ses extrémités serait contenue dans spq. Donc toute demi-droite issue d'un point x intérieur au segment pq et passant par s ne rencontre P qu'un seule fois (en s). Par le théorème de Jordan, et puisque la droite est extérieure à P à l'infini, le point x est intérieur à P.
- Sinon, soit r un sommet de P intérieur au triangle spq et qui minimise la distance à la droite pq. On montre aisément que le segment sr est une diagonale de P.

□

Exercice 3.2 *Compléter la preuve précédente en indiquant en particulier où intervient l'hypothèse sur le nombre minimal (4) de sommets.*

Théorème 3.3 *Tout polygone admet une triangulation.*

Preuve : Soit P un polygone et soit D un ensemble de diagonales de P d'intérieurs disjoints qui soit maximal pour l'inclusion. Toute région bornée du graphe plan $P \cup D$ est nécessairement un triangle ; dans la négative le lemme précédent contredirait la maximalité de D. □

On montre par récurrence sur n que toute triangulation d'un polygone à n sommets a exactement $n-2$ triangles et $n-3$ diagonales.

Exercice 3.4 *La preuve de Lennes pour l'existence d'une triangulation est proche de la précédente quoique légèrement différente. Soit spq un triangle et soit R un ensemble de points intérieurs à ce triangle. Montrer qu'il existe $r \in R$ tel que $rsp \cap R = \{r\}$ (ici rsp désigne le bord et l'intérieur du triangle). Compléter la preuve de Lennes.*

3.2 Algorithmes

Si un polygone P est décrit sous forme d'une liste doublement chaînée de sommets, la recherche d'une diagonale selon la preuve du lemme 3.1 prend un temps $O(n)$, où $n = |P|$ est le nombre de sommets de P. On en déduit aisément un algorithme de triangulation de complexité quadratique. Nous allons voir deux algorithmes plus efficaces.

Théorème 3.5 *Il existe un algorithme qui triangule tout polygone à n sommets en temps $O(n \log n)$ et espace $O(n)$.*

Nous proposons ci-dessous deux preuves – c'est à dire deux algorithmes – pour ce théorème. Elles sont respectivement décrites dans les articles suivants :

- A theorem on polygon cutting with applications. B. Chazelle. In Proc. IEEE Sympos. Found. Compu. Sci., pp. 339-349, 1982.

- Triangulating a simple polygon. M.R. Garey, D. S. Johnson, F. P. Preparata and R. E. Tarjan. Inform. Process. Lett., 7 :175-179, 1978.

3.2.1 Algorithme diviser pour régner

L'algorithme de Chazelle consiste à calculer en temps linéaire une diagonale du polygone P qui coupe P en deux sous-polygones de tailles approximativement égales. En appliquant ce calcul de manière récursive à chacun des deux sous-polygones on obtient un algorithme de triangulation de complexité $O(|P| \log |P|)$.

On supposera que P est décrit sous forme d'une liste cyclique doublement chaînée de ses sommets dans l'ordre (cyclique) le long de P. Par la suite P désignera aussi bien un polygone que sa liste doublement chaînée. On supposera également disposer de la liste L doublement chaînée des sommets de P triés selon l'ordre lexicographique de leurs coordonnées ainsi que de la liste V des paires d'arêtes de P *verticalement visibles*, i.e. des paires d'arêtes pour lesquelles il existe un segment vertical intérieur à P et dont les extrémités sont respectivement intérieures à chacune de ces deux arêtes. En considérant la taille de la carte des trapèzes (cf. section 6.2), il est facile de voir que la liste V a une taille linéaire en fonction de $|P|$.

Théorème 3.6 (du polygon cutting, Chazelle 1982) *Soit P un polygone à n sommets et soient L et V les listes associées (respectivement des sommets triés selon l'ordre*

lexicographique des coordonnées et des paires d'arêtes verticalement visibles). Il existe un algorithme de complexité $O(n)$ pour calculer une diagonale de P qui coupe P en deux polygones P_1 et P_2 tels que

$$|P_1|, |P_2| \leq \lceil \frac{2}{3} n \rceil + 1.$$

De plus, les listes P_i, L_i et V_i relatives à chacun des deux polygones pour $i = 1, 2$ peuvent être calculées en temps $O(n)$ également.

On montre dans un premier temps qu'il existe un segment vertical intérieur à P et qui le coupe en deux polygones de tailles approximativement égales.

Lemme 3.7 *Soit P un polygone à n sommets. Il existe un segment vertical pq (i.e. $x_p = x_q$) intérieur à P et intersectant P en ses deux extrémités p et q exactement de sorte que les deux composantes de $P \setminus \{p, q\}$ contiennent chacune au plus $\lceil \frac{2}{3} n \rceil$ sommets de P.*

Preuve : On considère la décomposition trapézoïdale de P (cf. section 6.2). On suppose dans un premier temps que P est en position générique, c'est à dire que tous les sommets de P ont des abscisses distinctes. Il suit que chaque trapèze de la décomposition intérieur à P est bordé par exactement un sommet de P sur sa gauche et un sommet de P sur sa droite. Soit T le graphe plan obtenu en reliant par un segment chaque paire de sommets incidents à un même trapèze (il y a donc un segment par trapèze). T est connexe (utiliser par exemple la connexité du graphe d'adjacence des trapèzes) et acyclique (utiliser l'acyclicité du graphe d'adjacence des trapèzes), i.e. que T est un arbre. De plus, l'hypothèse de position générique montre que chaque sommet est incident à trois trapèzes au plus et donc que le degré des sommets de T est au plus trois.

Remarquons qu'on peut associer à chaque arête a de T un segment vertical intérieur à P et intersectant P en ses extrémités. Le nombre de sommets des deux lignes polygonales de P coupées par ces extrémités est précisément le nombre de sommets de chaque sous arbre de $T - a$. Le lemme suivant permet donc de conclure.

Dans le cas non-générique, on perturbe les sommets en tournant de manière infinitésimale le polygone afin de distinguer toutes les abscisses des sommets de P. Cette rotation est elle-même simulée en considérant l'ordre lexicographique sur les paires (abscisses, ordonnées) des coordonnées des sommets. □

Lemme 3.8 *Soit T un arbre à n sommets, $n \geq 2$. On suppose que chaque sommet de T est de degré au plus 3. Alors il existe une arête a de T telle que chaque composante de $T - a$ possède au plus $\lceil \frac{2n}{3} \rceil$ sommets.*

Preuve : Soit a une arête de T. Si une composante C de $T - a$ est telle que $|C| < \lfloor \frac{n}{3} \rfloor$ alors il existe une arête b de T telle que

1. ou bien une composante K de $T - b$ vérifie

$$|C| < |K| < \lfloor \frac{n}{3} \rfloor$$

2. ou bien chaque composante de $T - b$ est de taille au moins $\lfloor \frac{n}{3} \rfloor$.

En effet, soit C' la composante complémentaire de C dans $T - a$ et x le sommet incident à C' et a. Par les hypothèses sur T, x est de degré $d \leq 2$ dans C'.
- On ne peut avoir $d = 0$ car dans ce cas on aurait $|C'| = 1 > \lceil \frac{2n}{3} \rceil$, en contradiction avec $n \geq 2$.
- Si $d = 1$, on choisit pour b l'unique arête incidente x dans C'. Alors $T - b$ a une composante K (de fait $C + a$) de taille $|C| + 1$. On se retrouve alors dans le cas 1 ou 2 ci-dessus selon que $|C| + 1$ est respectivement strictement inférieur ou égal à $\lfloor \frac{n}{3} \rfloor$.
- Sinon $d = 2$. Soient a_1, a_2 les deux arêtes incidentes à x dans C' et soit C_1 (resp. C_2) la composante de $C' - a_1$ (resp. de $C' - a_2$) qui n'est pas incidente à x. Si $|C_1| > \lceil \frac{2n}{3} \rceil$ alors on se retrouve dans le cas 1 en choisissant $b = a_1$ (et $K = C + a + C_2$). De même en choisissant $b = a_2$ si $|C_2| > \lceil \frac{2n}{3} \rceil$. On peut donc supposer $|C_1| \leq \lceil \frac{2n}{3} \rceil$ et $|C_2| \leq \lceil \frac{2n}{3} \rceil$. Puisqu'on ne peut avoir à la fois $|C_1| < \lfloor \frac{n}{3} \rfloor$ et $|C_2| < \lfloor \frac{n}{3} \rfloor$, on a $|C_1| \geq \lfloor \frac{n}{3} \rfloor$ ou $|C_2| \geq \lfloor \frac{n}{3} \rfloor$. On se retrouve dans le cas 2 en choisissant $b = a_1$ dans le premier cas et $b = a_2$ dans le second cas.

La preuve du lemme est alors terminée par récurrence sur |C|. □

Preuve du théorème 3.6 : On sait d'après le lemme 3.7 qu'il existe une verticale dont les extrémités coupent P en deux lignes polygonales contenant chacune au plus $\lceil \frac{2n}{3} \rceil$ sommets. En parcourant la liste V des paires d'arêtes verticalement visibles on trouvera donc nécessairement une paire (a, b) telle que tout segment vertical de visibilité entre les deux arêtes a et b coupe P comme ci-dessus. Fixons un sommet s de P et indexons les sommets de P de 0 à $n - 1$ dans l'ordre direct le long de P à partir de s. À partir de ces indices on peut calculer en temps constant la longueur de la ligne polygonale entre deux sommets d'indices i et j : en incluant les deux sommets et en considérant la ligne de i vers j dans le sens direct, cette longueur vaut $j - i + 1$ si $j \geq i$ et $n - j + i - 1$ sinon. On peut donc tester en temps constant si une paire d'arêtes convient et de ce fait déterminer (a, b) en temps linéaire.

Il n'est en général pas possible de relier deux des extrémités de a et de b par une diagonale car celle-ci peut recouper P. Considérons le quadrilatère Q formé par a, b et les deux segments c et d reliant les extrémités de a et b situés d'un même côté d'une verticale de visibilité entre a et b. Le quadrilatère Q forme bien un polygone (simple) de par l'existence d'un segment de visibilité qui sépare c et d. Soit P_c (resp. P_d) la sous-ligne polygonale de P bordés par les extrémités de c (resp. de d) et ne contenant ni a ni b. L'objectif est de calculer une diagonale entre un sommet de P_c intérieur à Q et un sommet de P_d intérieur à Q de sorte que cette diagonale sépare P comme voulu. Pour cela on considère l'ensemble S_c (resp. S_d) des composantes de $P \cap \overline{IntQ}$ qui s'appuient sur c (resp. sur d). On pose $C_c = Conv(S_c)$ et $C_d = Conv(S_d)$

Affirmation I : C_c et C_d sont disjointes. De plus, les sommets de C_c (resp. de C_d) sont les sommets de P_c (resp. de P_d).

Preuve de l'affirmation I : Par hypothèse, il est facile de voir – à l'aide du théorème de Jordan – que S_c et S_d sont séparées dans Q par un segment vertical. Il en est donc de même de leurs enveloppes convexes. Par ailleurs, toujours à l'aide du théorème de Jordan, on montre que toute région bordée par une composante de $S_c \cap P_d$ et un segment de c

est contenue dans une région bordée par une composante de $S_c \cap P_c$ et un segment de c. On en déduit que $C_c = Conv(S_c \cap P_c)$. Donc C_c est l'enveloppe convexe des sommets de $S_c \cap P_c$ qui comprend les sommets de P_c inclus dans S_c et les extrémités des composantes de S_c. Or ces sommets sont tous sur c donc dans l'enveloppe convexe des extrémités de c qui sont des sommets de P_c. Un raisonnement analogue montre que les sommets de C_d sont des sommets de P_d. □

L'affirmation précédente permet de sélectionner en temps $O(n)$ un sous-ensemble A des sommets de P tel que $C_c = Conv(A)$: il suffit de parcourir P_c et de retenir les sommets de P_c compris entre deux intersections successives de P_c avec c, lorsque P_c entre dans Q à la première intersection. On peut extraire de L la sous-liste L_A, triée selon l'ordre lexicographique des coordonnées, des sommets de A. On obtient finalement C_c en temps linéaire à partir de L_A par l'algorithme classique de balayage **??**.[1] De manière analogue on calcule C_d en temps linéaire.

On considère maintenant le polygone Q' formé des arêtes a et b et des deux chaînes concaves $C'_c = C_c - c$ et $C'_d = C_d - d$. Notons que Q' est bien une ligne polygonale simple d'après l'affirmation I. Notre but est de montrer que dans toute triangulation $\mathcal{T}$ de Q' l'une des arêtes de $\mathcal{T}$ fournit une diagonale qui sépare P comme voulu. On remarque tout d'abord que tout triangle de $\mathcal{T}$ contient nécessairement une arête de C'_c ou bien de C'_d. En effet, C'_c et C'_d étant concaves, un tel triangle ne peut avoir deux sommets non adjacents sur une même de ces deux chaînes. Le dual de $\mathcal{T}$ est donc une chaîne simple, ce qui permet d'ordonner les diagonales de $\mathcal{T}$ de la première, incidente au même triangle que a, à la dernière, incidente au même triangle que b. On note $\delta_1, \delta_2, \ldots \delta_k$ ces diagonales, on pose $\delta_{k+1} = b$ et pour $i = 1, \ldots, k-1$, on note γ_i la troisième arête du triangle de $\mathcal{T}$ bordé par δ_i et δ_{i+1}. Donc γ_i est une arête de C'_c ou de C'_d et on note Γ_i la sous-chaîne de respectivement P_c ou P_d joignant les extrémités de γ_i. Pour chaque diagonale δ_i de $\mathcal{T}$, on note enfin Δ_i la sous-chaîne de P contenant a et joignant les extrémités de δ_i.

Affirmation II : Une des arêtes de $\mathcal{T}$ (soit une diagonale soit une arête de Q') est une diagonale de P qui coupe P en deux lignes polygonales (extrémités inclues) de taille au plus $\lceil \frac{2}{3}n \rceil + 1$.

Preuve de l'affirmation II : Supposons qu'une arête γ_i de C'_c ou de C'_d ne soit pas une arête de P (et soit donc une diagonale de P) et que

$$|\Gamma_i| \geq \lfloor \frac{n}{3} \rfloor + 1.$$

On a alors, en notant Γ'_i l'autre chaîne de P joignant les extrémités de γ_i
– d'une part : $|\Gamma_i| \leq \max\{|P_c|, |P_d|\} \leq \lceil \frac{2n}{3} \rceil$,
– d'autre part : $|\Gamma_i| + |\Gamma'_i| = n + 2$, d'où $|\Gamma'_i| \leq \lceil \frac{2}{3}n \rceil + 1$.
L'affirmation est donc vérifiée en choisissant γ_i comme diagonale de P.

Supposons maintenant à l'inverse que pour toute arête γ_i de C'_c ou de C'_d on ait

$$|\Gamma_i| \leq \lfloor \frac{n}{3} \rfloor.$$

[1] Chazelle utilise un autre argument. Il extrait de S_c une ligne polygonale simple joignant les extrémités de c et dont l'enveloppe convexe est C_c Cette ligne est constituée de composantes de S_c et de segments de c. Il utilise ensuite l'algorithme de complexité linéaire pour calculer l'enveloppe convexe d'une ligne polygonale simple.

On a en particulier

$$|\Delta_1| = |\Gamma_1 + a| \leq \lfloor \frac{n}{3} \rfloor + 1.$$

Par ailleurs si pour un certain $i \in [1, k]$ on a $|\Delta_i| \leq \lfloor \frac{n}{3} \rfloor$ alors δ_i n'est pas la dernière diagonale de $\mathcal{T}$ (i.e. $i < k$) et

$$|\Delta_i| < |\Delta_{i+1}| \leq \lceil \frac{2}{3} n \rceil + 1.$$

En effet, on a

$$|\Delta_{i+1}| = |\Delta_i + \Gamma_i| = |\Delta_i| + |\Gamma_i| - 1 \leq 2 \lfloor \frac{n}{3} \rfloor - 1.$$

Alors que $|\Delta_k| \geq \lceil \frac{2}{3} n \rceil + 1$ (faire un raisonnement analogue à la majoration de $|\Delta_1|$). On conclut par récurrence sur k que l'une au moins des diagonales δ_i vérifie

$$\lfloor \frac{n}{3} \rfloor + 1 \leq |\Delta_i| \leq \lceil \frac{2}{3} n \rceil + 1.$$

Ceci permet également de confirmer l'affirmation en choisissant δ_i comme diagonale de P. □

Affirmation III : Q' peut être triangulé en temps linéaire.

Preuve de l'affirmation III : D'après la preuve de l'affirmation II, on peut trianguler Q' de manière incrémentale en calculant chaque diagonale δ_{i+1} en fonction de la diagonale δ_i calculée précédemment : en notant p_c et p_d les extrémités de δ_i et $p_c q_c$ et $p_d q_d$ les arêtes de Q' respectivement incidentes à p_c et p_d et "au dessus" de δ_i, alors on a soit $\delta_{i+1} = p_c q_d$ soit $\delta_{i+1} = p_d q_c$. Il suffit de tester si q_d (resp. p_c) est au dessous de la droite $p_c q_c$ (resp. $p_d q_d$) pour savoir si $p_c q_d$ (resp. $p_d q_c$) est une diagonale. Il se peut que les deux le soient, auquel cas l'une ou l'autre convient puisque dans les deux cas on se retrouve dans une configuration où la partie de Q' au dessus de δ_{i+1} est constituée de deux chaînes concaves reliées par deux segments, ce qui permet d'appliquer la récurrence. □

La conjonction des affirmations II et III permet de conclure la première partie du théorème. Il reste à vérifier, en appelant P_1 et P_2 les deux polygones coupés par la diagonale δ trouvée, que les listes P_i, L_i et V_i relatives au polygone P_i pour $i = 1, 2$ peuvent être calculées en temps $O(n)$ également. C'est clair pour les listes P_i et L_i (ces listes contiennent plus précisément des pointeurs bidirectionnels sur un tableau des sommets fixé une fois pour toute. On peut associer à chacun des sommets du tableau un drapeau qui permet de sélectionner dans une première passe les sommets qui nous intéressent). Pour la liste V_i il suffit de remarquer qu'elle est constituée d'une part des paires d'arêtes de V qui sont dans P_i et d'autre part des paires (a, δ) pour chaque paire (a, b) de V dont la visibilité est obstruée par δ et telle que a est une arête de P_i (et donc b n'est pas une arête de P_i). En parcourant V, on peut ainsi construire V_i de la manière suivante. Pour chaque paire (a, b) de V :

1. si $a \in P_i$ et $b \in P_i$, alors on place (a, b) dans V_i,

2. sinon, si a et b ne sont pas dans le même polygone, disons $a \in P_i$, $b \notin P_i$, et si les projections verticales sur l'axe des abscisses des arêtes a, b et δ ont une intersection non vide, alors on place (a, δ) dans V_i après avoir vérifié que cette paire n'était pas déjà présente dans V_i. Cette dernière vérification s'obtient en temps constant en marquant au fur et à mesure les arêtes a de P telles que (a, δ) est dans V_i.

□

Preuve du théorème 3.5 : Soit P donc un polygone à n sommets. L'algorithme consiste à appliquer récursivement le théorème du polygon cutting : la triangulation de P est l'union des triangulations des polygones P_1 et P_2 obtenus par le théorème 3.6.

Pour initialiser l'algorithme il faut construire la liste L des sommets de P triés selon l'ordre lexicographique de leurs coordonnées ; ce qui prend un temps $O(n \log n)$ et une place linéaire suivant tout algorithme de tri standard. Il faut construire également la liste V des paires d'arêtes de P *verticalement visibles* que l'on obtient en temps linéaire en parcourant la carte des trapèzes de P (cf. section 6.2). Cette carte a elle-même une taille linéaire (lemme 6.2) et peut être construite en temps $O(n \log n)$ par un algorithme randomisé (cf. section 6.2) ou non (cf. section 6.1).

Soit $C(n)$ la complexité maximale de la triangulation de tout polygone de taille n. On peut écrire

$$C(n) \leq kn + \max_{\substack{n_1+n_2=n+2 \\ n_1,n_2 \leq \lceil \frac{2}{3}n \rceil + 1}} \{C(n_1) + C(n_2)\}$$

pour un certain $k > 0$. Montrons que $C(n) = O(n \log n)$. Soit α tel que $2/3 < \alpha < 1$. Choisissons N assez grand pour que $n > N \implies \lceil \frac{2}{3}n \rceil + 1 < \alpha n$ et $\frac{n}{2} \log \frac{1}{\alpha} > 2 \log n$. Choisissons ensuite K suffisamment grand pour que $n \leq N \implies C(n) \leq Kn \log n$ et pour que $K \log \frac{1}{\alpha} > 2k$.

Pour $n \leq N$ on a donc par hypothèse $C(n) \leq Kn \log n$. Supposons par récurrence $C(m) \leq Km \log m$ pour m inférieur à un certain $n > N$. Pour $n_1, n_2 \leq \lceil \frac{2}{3}n \rceil + 1$ tels que $n_1 + n_2 = n + 2$ on a

$$\begin{aligned} kn + C(n_1) + C(n_2) &\leq kn + Kn_1 \log n_1 + Kn_2 \log n_2 \leq kn + K(n+2) \log(\alpha n) \\ &\leq Kn \log n + 2K \log n + (k - K \log \frac{1}{\alpha})n \\ &\leq Kn \log n + K(2 \log n - \frac{n}{2} \log \frac{1}{\alpha}) \leq Kn \log n \end{aligned}$$

Ce qui permet de conclure $C(n) \leq Kn \log n$. □

3.2.2 Algorithme par décomposition en polygones monotones

L'algorithme de triangulation de Garey et al. se compose de deux étapes. Dans un premier temps le polygone à trianguler est décomposé en polygones plus simples appelés polygones *monotones*. Cette étape prend un temps $O(n \log n)$. Ces polygones monotones sont ensuite triangulés en temps linéaire selon une technique appropriée. Au total on obtient donc une complexité équivalente à l'algorithme diviser pour régner de Chazelle.

Polygones monotones

Définition 3.9 *On considère une direction du plan qu'on appelle* verticale. *La direction orthogonale est dite* horizontale. *La* hauteur *d'un point est sa projection horizontale sur la verticale. Une ligne polygonale L est dite (strictement)* monotone *si la* hauteur *de la séquence des sommets de L est (strictement) monotone. Dit autrement L est monotone si toute droite horizontale coupe L en au plus une composante, réduite à un point dans le cas strict.*

Un polygone P est dit (strictement) monotone *s'il est la réunion de deux lignes polygonales (strictement) monotones ayant seulement leurs extrémités en commun. Dit autrement un polygone P est monotone (resp. strictement monotone) si toute droite horizontale coupe P en au plus deux composantes (resp. au plus deux points).*

Un sommet intérieur (i.e. qui n'est pas une extrémité) à une ligne polygonale ou à un polygone est dit maximum, (resp. minimum) *(strict) si ces deux sommets voisins sont (strictement) en dessous (resp. au dessus) de la droite horizontale passant par ce sommet. On appelle* extremum *(strict) un sommet qui est soit maximum (strict) soit minimum (strict).*

On vérifie aisément qu'un sommet n'est pas un extremum si et seulement si sa hauteur est strictement comprise entre celles de ses deux sommets voisins. Par suite :

Lemme 3.10
- *une ligne polygonale ayant au moins 3 sommets est strictement monotone si et seulement si aucun de ses sommets intérieurs n'est extremum.*
- *Un polygone est strictement monotone si et seulement si il a exactement deux extrema.*

Définition 3.11 *Un sommet d'un polygone est dit* réflexe *si l'angle* intérieur au polygone *formé par les deux arêtes incidentes au sommet est strictement plus grand que π. Une sous-chaîne d'un polygone est dite* concave *si ses sommets intérieurs sont réflexes.*

Lemme 3.12 *Un polygone sans extremum réflexe est monotone.*

Preuve : Soit P un polygone sans extremum réflexe. Soient p et q des sommets de P de hauteur respectivement minimale et maximale. Les sommets p et q coupent P en deux lignes polygonales P_G et P_D telles que P_G est à gauche de P_L. Supposons par l'absurde que P n'est pas monotone. Alors par définition, P_G ou P_D n'est pas monotone. On suppose sans perte de généralité que P_G n'est pas monotone. Il existe donc une droite horizontale h coupant P_G en deux composantes au moins. On note h^+ et h^- les demi-plans ouverts respectivement au dessus et au dessous de h. Par connexité de P_G, l'une des composantes, C, de $h^+ \cap P_G$ ou de $h^- \cap P_G$ a ses deux extrémités dans h. On note u et v ces deux extrémités avec u à gauche de v. Supposons à nouveau sans perte de généralité $C \subset h^+$. J'affirme que

(A) le long de C, l'intérieur de P est situé du même côté que l'intérieur du polygone délimité par C et le segment uv de h.

Puisque P_G est à gauche de P_D, l'intérieur de P est à droite de P_G lorsque P_G est parcourue du bas vers le haut (i.e. de p vers q). Par conséquent u est avant v dans ce parcours. Soit D la composante de $P_G \setminus C$ joignant v à q. J'affirme que

(B) le sommet le plus à gauche parmi les sommets de hauteur minimale de D est extremum réflexe.

Cette dernière contradiction permet de conclure la monotonicité de P_G et donc de P. Il reste à montrer les affirmations (A) et (B).

Pour (A), on considère le sommet x de C le plus à droite parmi les sommets de hauteur maximale. Les directions des deux arêtes d'origine x et la direction horizontale $\vec{h}$ vers la droite sont donc deux à deux distinctes. Soit a l'arête issue de x dont la direction suit celle de $\vec{h}$ dans le sens indirect (le sens des aiguilles d'une montre). On note b la seconde arête issue de x. Comme x est extremum, il ne peut être réflexe, ce qui montre que l'intérieur de P est entre b et a dans le sens direct, ou encore à droite de a. Montrons que c'est également le cas pour l'intérieur de la courbe de Jordan $uv \cup C$. Pour cela, on note C_a et C_b les deux composantes de $C \setminus \{x\}$ contenant respectivement a et b et on note $w \in \{u, v\}$ l'extrémité de C_a autre que x. Il est clair qu'on ne peut avoir $w = u$, sinon la courbe simple S formée de C_a, de la demi-droite horizontale à droite de x et de la demi-droite horizontale à gauche de w formerait une courbe de Jordan qui ne rencontre pas C_b. Or l'intérieur de b est au dessus de S et l'extrémité v de C_b est au dessous de S, ce qui contredit la connexité de C_b. Donc $w = v$. Comme l'intérieur de $uv \cup C$ est au dessus et donc à droite de vu, il en est de même pour C_a, i.e. l'intérieur de $uv \cup C$ est à droite de a.

Un raisonnement analogue permet de montrer que le sommet spécifié dans (B) qui est évidemment extremum est également réflexe. □

Décomposition en polygones monotones

Théorème 3.13 *Il existe un algorithme de complexité $O(n \log n)$ pour décomposer tout polygone à n sommets en polygones v-monotone sans ajouter aucun sommet.*

Preuve : Considérons la décomposition trapézoïdale d'un polygone P obtenue par cloisonnement horizontal. On suppose le polygone en position générale, i.e. deux sommets distincts ont des ordonnées distinctes. Chaque trapèze est donc incident à exactement deux sommets de P, un sommet supérieur sur le côté horizontal supérieur du trapèze et un sommet inférieur sur le côté horizontal inférieur du trapèze. On ajoute une diagonale joignant ces deux sommets si le sommet supérieur est un minimum réflexe et/ou si le sommet inférieur est un maximum réflexe. On obtient ainsi une décomposition de P en polygones. On vérifie qu'aucun sommet de P ne peut être extremum réflexe dans les polygones qui lui sont incidents. Il suit du lemme 3.12 que ces polygones sont tous monotones. Le cas non général où plusieurs sommets peuvent posséder une même ordonnée se traite en simulant une perturbation par rotation à l'aide de l'ordre lexicographique sur les paires (ordonnée, abscisse).

Notons que la décomposition trapézoïdale peut s'obtenir en temps $O(n \log n)$ et que l'ajout de chaque diagonale s'obtient en temps constant par diagonale (en utilisant une

structure de carte planaire en demi-arêtes), ce qui achève la démonstration. □

Exercice 3.14 *Décrire un algorithme de complexité $O(n \log n)$ pour la décomposition d'un polygone P en polygones monotones qui n'utilise pas à proprement parler la décomposition trapézoïdale de P, mais seulement un balayage des sommets de P.*

Triangulation des polygones monotones

Théorème 3.15 *Il existe un algorithme de complexité linéaire pour trianguler un polygone v-monotone.*

Preuve : Soit P un polygone strictement monotone. En temps linéaire on peut couper P en deux chaînes monotones. Les sommets sur chaque chaîne sont naturellement triés selon leur ordonnée. La fusion (en temps linéaire) de ces deux listes permet d'obtenir la liste V des sommets de P triés selon leur ordonnée. On effectue un balayage des sommets de haut en bas. Au cours du balayage une partie de l'intérieur de P est triangulée et une autre P_i forme un polygone strictement monotone que l'on doit trianguler. Soit σ_i le sommet maximum de P_i et soient G_i et D_i les deux chaînes monotones maximales de P_i. On stocke les sommets balayés dans une pile Π de manière à conserver l'invariant suivant : (i) les sommets dans Π forment une sous-chaîne concave de sommets réflexes de P_i issue de σ_i et (ii) le sommet σ_i est également incident dans P_i à un sommet ν_i plus bas que les sommets de Π. En particulier, si les sommets de Π sont dans G_i (resp. D_i) alors ν_i est dans D_i (resp. G_i). Au départ Π est initialisée avec les deux premiers sommets de V (i.e. le sommet maximal de $P_0 = P$ et le sommet juste au dessous).

Soit s le nouveau sommet balayé dans la liste V.

1. Si s forme une chaîne concave avec Π (i.e. le dernier sommet de Π est réflexe) alors on empile s. Les invariants (i) et (ii) sont maintenus.
2. Si $s = \nu_i$ et si s n'est pas le sommet minimal de P alors on relie s par des segments à chacun des sommets de Π, hormis σ_i (qui est déjà relié à s). Notons que ces segments sont des diagonales de P_i (et donc de P) car aucune arête entre deux sommets de Π ne peut couper une telle diagonale (par concavité de Π) ni aucune autre arête de P_i par monotonie de Π. On a ainsi triangulé une partie supérieure de P_i. Le reste constitue le polygone P_{i+1}. On vide ensuite Π et on ré-insert son dernier sommet, qui devient le sommet σ_{i+1}, puis le sommet s ; ces deux sommets formant les deux plus hauts sommets de P_{i+1}. On définit également ν_{i+1} comme le sommet suivant Π le long de P_i. Clairement les invariants (i) et (ii) sont rétablis.
3. Si s est incident au dernier sommet s_ℓ de Π mais ne forme pas une chaîne concave avec Π (i.e. s_ℓ est convexe dans P_i) alors on relie s par des segments aux derniers sommets $s_k, s_{k+1}, \ldots, s_{\ell-1}$ de Π, hormis le tout dernier s_ℓ auquel il est déjà relié, de sorte que la droite ss_k est support pour Π (i.e. les sommets de Π sont d'un même côté de cette droite) mais que la droite ss_{k+1} ne l'est pas. À nouveau ces segments sont des diagonales de P_i car aucune des deux chaînes G_i et D_i ne peut recouper le segment ss_k du fait de leur monotonie. On a ainsi triangulé une partie de P_i. Le reste constitue le polygone P_{i+1}. On dépile alors les sommets $s_{k+1}, \ldots, s_\ell$ de Π

et insert le sommet s. Le fait que la droite ss_k soit support de Π montre que Π contient bien une chaîne concave et que les invariants (i) et (ii) sont rétablis.

Lorsqu'on balaye le sommet minimal de P on se retrouve dans la dernière des trois situations ci-dessus et la triangulation qui suit achève la triangulation de P. Pour chaque sommet balayé les opérations effectuées ci-dessus se décomposent dans chacun des trois cas en un nombre constant d'opérations élémentaires auquel s'ajoute un nombre d'opérations proportionnel au nombre de diagonales ajoutées à la triangulation. Comme il y a un nombre linéaire de diagonales et que l'on balaye un nombre linéaire de sommets, le coût total de la triangulation est linéaire. □

3.2.3 Application au problème de la galerie d'art

Le problème communément attribué à Victor Klee en 1973 est le suivant : étant donné une galerie d'art dont le sol a la forme d'un polygone, combien de gardiens (ou caméras) fixes suffisent à garder la galerie ? On sous-entend que chaque gardien peut regarder dans toutes les directions autour de lui.

De manière plus géométrique, soit P un polygone du plan et $x \in \overline{IntP}$ un point intérieur (au sens large) à P. La *zone de visibilité* de x dans P est l'ensemble des points intérieurs à P et visibles depuis x dans P. C'est encore

$$V_P(x) = \{y \in \overline{IntP} \,|\, xy \subset \overline{IntP}\}$$

où xy dénote le segment joignant x et y. Un ensemble $X \subset P$ *couvre* P si

$$P = \cup_{x \in X} V_P(x)$$

i.e. si l'union des zones de visibilités des points de X recouvre P. Le problème de la galerie d'art revient donc à chercher un ensemble X de taille minimale couvrant P. Le problème de Klee était plus précisément de trouver la taille minimale $g(n)$ telle que *tout* polygone à n sommets est couvert par un ensemble de taille $g(n)$.

Théorème 3.16 (de la galerie d'art, Chvátal 1973)

$$g(n) = \lfloor n/3 \rfloor.$$

La preuve suivante due à Fisk en 1978 utilise la notion de coloriage. Un *coloriage* d'un ensemble E par une ensemble C est une application de E dans C. Si le cardinal de C est k on parle d'un *k-coloriage* de E.

Preuve du théorème : $g(n) \leq \lfloor n/3 \rfloor$: soit P un polygone à n sommets et $\mathcal{T}$ une triangulation de P. L'algorithme suivant construit un 3-coloriage des sommets de P tel que tout triangle de $\mathcal{T}$ est tricolore (ses trois sommets ont des couleurs 2 à 2 distinctes) :

Choisir une arête a de P et colorier ses deux sommets, l'un en bleu et l'autre en blanc. Cette arête est incidente à un unique triangle de $\mathcal{T}$, ce qui détermine la couleur de son troisième sommet, disons rouge. Si l'arête (rouge, blanc) de ce triangle est une diagonale,

colorier récursivement la partie de P coupée par cette diagonale et ne contenant pas a. Faire de même avec l'arête (rouge, bleu).

Remarquons alors que l'ensemble des sommets de P ayant la couleur la moins fréquente dans un tel 3-coloriage est de cardinal au plus $\lfloor n/3 \rfloor$ et couvre P, puisque couvre tout triangle de $\mathcal{T}$.

$g(n) \geq \lfloor n/3 \rfloor$: Pour tout k,on construit un polygone en forme de peigne de taille $3k$, ayant k dents, qui ne peut être couvert par moins de k points. Considérons pour cela un triangle t ayant un côté horizontal de longueur 1 et k copies $t_1, t_2, \ldots, t_k$ de t successivement translatées horizontalement de 2 unités. Ces copies sont donc disjointes et forment les dents du peigne. On considère également un trapèze dont les bases sont horizontales : l'une joint un sommet du côté horizontal de t_1 avec un sommet du côté horizontal de t_k et l'autre joint un point intérieur à un côté non horizontal de t_1 avec un point intérieur à un côté non horizontal de t_k. Notre peigne est le bord de l'union de ce trapèze avec les k triangles $t_1, t_2, \ldots, t_k$. Puisque les triangles sont disjoints, aucun point du peigne ne peut couvrir simultanément les deux pointes de deux dents du peigne. Il faut donc au minimum k points pour couvrir le peigne. □

L'algorithme de coloriage de la preuve du théorème est clairement de complexité linéaire. Compte tenu de l'existence d'un algorithme de triangulation de complexité linéaire, ceci permet de trouver en temps linéaire, pour un polygone donné, un ensemble couvrant de taille minimale dans le cas le pire. Par contraste, trouver la taille minimale de tout ensemble couvrant pour un polygone précis est un problème NP-difficile (Aggarwal, 1984).

Références :

- `http ://valis.cs.uiuc.edu/∼sariel/teach/2004/b/webpage/lec/23_triang.pdf`
- `http ://valis.cs.uiuc.edu/∼sariel/teach/2004/b/webpage/lec/24_triang_II.pdf`
- On pourra consulter l'État de l'art
Art Gallery and Illumination Problems. J. Urratia. Chap. 22 in "Handbook of Computational Geometry". Edited by J. R. Sack and J. Urrutia

Chapitre 4

Recherche monodimensionnelle

4.1 Dictionnaires

Un dictionnaire est une structure de données permettant de rechercher, insérer ou supprimer des données. Chaque donnée est supposée posséder une clé qui l'identifie. Cette clé doit appartenir à un univers totalement ordonné (typiquement des entiers). Pour ranger les données on se sert de leur clé. Par la suite on ne s'intéresse qu'aux clés, les "véritables" données pouvant être obtenues à partir des clés à l'aide d'un pointeur par exemple. D'autres opérations telles que la recherche de la clé minimale ou maximale, de la clé suivant ou précédant une clé donnée sont possibles. La fusion et la scission de dictionnaires sont également des opérations courantes.

Classiquement, mais pas uniquement (voir plus bas) les dictionnaires sont représentés à l'aide d'arbres. Ces arbres peuvent être binaires (comme pour les arbres *AVL* ou *bicolores)* ou non (*arbres a-b*).

Références :

http ://www.cs.sunysb.edu/∼algorith/

http ://www.nist.gov/dads/

Introduction to Algorithms. Cormen, Rivest, Leiserson and Stein, M.I.T. PRESS, second edition 2001. Version française chez Dunod, 1994.

STL (http ://www.sgi.com/tech/stl/)

LEDA (http ://www.mpi-sb.mpg.de/LEDA/MANUAL/MANUAL.html)

4.1.1 Arbres binaires de recherche

Définition 4.1 *Un* arbre binaire de recherche *est un arbre binaire dont les noeuds possèdent des clés rangées dans l'ordre infixe, i.e. un parcours dans l'ordre infixe de l'arbre (sag(b).racine(b).sad(b)) visite les clés dans l'ordre croissant.*

Propriété : L'ensemble des arbres binaires de recherche est stable par les opérations de

rotation.

Note : parfois on ne range les clés qu'aux noeuds externes et un noeud interne contient une valeur intermédiaire entre celles de son sous-arbre gauche et de son sous-arbre droit.

4.2 Structures randomisées

Les structures classiques telles que les arbres AVL, bicolores ou a-b sont des structures déterministes. Les performances des opérations de dictionnaires ne dépendent ni des données ni de l'ordre de construction de ces structures. Les algorithmes de construction et de modification sont cependant relativement compliqués. Les structures randomisées telles que les *skip lists* ou les *treaps* utilisent des générateurs aléatoires dans leur construction et permettent d'obtenir des performances *en moyenne* équivalentes à celles des structures déterministes. Les constructions sont généralement plus simples.

4.2.1 Skip list

Une structure de *skip list* sur un ensemble M de n clés est un dictionnaire sur M construit de la manière suivante :

On tire *au hasard et de manière indépendante* chaque clé de M avec une probabilité $1/2$. On obtient ainsi un sous-ensemble M_2 de M avec lequel on recommence la procédure de tirage. On continue ainsi jusqu'à obtenir l'ensemble vide. On obtient finalement une *gradation* de M, c'est à dire une suite décroissante de sous-ensembles de M :

$$M = M_1 \supseteq M_2 \supseteq \ldots \supseteq M_r \supset M_{r+1} = \emptyset$$

Une skip list sur une telle gradation s'obtient à partir des r listes triées de chaque M_i, appelé niveau, augmenté pour chaque clé de la liste M_i d'un pointeur vers l'occurrence de cette clé dans la liste M_{i-1} et d'un pointeur vers son successeur dans la liste M_i. On ajoute également un élément fictif minimal dans chaque liste que l'on relie entre-eux. On s'intéresse à la taille de cette structure et à la complexité des opérations de recherche, insertion, suppression en fonction du nombre n de clés. Puisque ces grandeurs dépendent de tirages aléatoires, on s'intéresse à leur valeur moyenne où l'on considère tous les tirages de clés indépendants. De plus, pour caractériser le fait que les grandeurs s'écartent très peu des valeurs moyennes (c.a.d. que la distribution des valeurs est bien localisée autour de la moyenne) on introduit la notation suivante :

Définition 4.2 *Soient $f(n)$ et $g(n)$ des variables aléatoires dépendant d'un paramètre n. On écrit $f = \tilde{O}(g)$ si $P(f(n) > cg(n)) < 1/p(n,c)$ où $p(n,c)$ est un polynôme en n dont le degré tend vers l'infini avec c.*

Intuitivement, un $\tilde{O}(g)$ est un $O(g)$ avec très forte probabilité.

Soit h_i la variable aléatoire qui donne le nombre de niveaux auxquels appartient la clé i. h_i suit une loi géométrique de paramètre $1/2$, d'où $P(h_i = k) = 1/2^k$ et $E(h_i) = 2$.

On pose $h = \max_{1 \leq i \leq n} h_i$ la hauteur d'une skip list.

Lemme 4.3 *$E(h) = O(\log n)$ et $h = \tilde{O}(\log n)$.*

Preuve :

$$P(h > k) = P((h_1 > k) \vee (h_2 > k) \vee \ldots \vee (h_n > k)) \leq \sum_{i=1}^{n} P(h_i > k) = n/2^k.$$

On en déduit pour tout $c > 0$ que $P(h > c \log n) \leq 1/n^{c-1}$, d'où $h = \tilde{O}(\log n)$. On a de plus (cf. lemme 1.9)

$$E(h) = \sum_{k=0}^{c\log n - 1} P(h > k) + \sum_{k=c\log n}^{\infty} P(h > k) \leq c \log n + n \sum_{k=c\log n}^{\infty} \frac{1}{2^k} = c \log n + \frac{2}{n^{c-1}}.$$

□

On note $t = \sum_{1 \leq i \leq n} h_i$ la taille d'une skip list.

Lemme 4.4 *La taille t d'une skip list sur n clés vérifie $E(t) = O(n)$ et $t = \tilde{O}(n)$.*

Preuve : Par linéarité de l'espérance, on a $E(t) = \sum_i E(h_i) = 2n$. En mettant bout a bout les tirages pour chaque clé, t s'interprète comme le nombre de tirages nécessaires pour obtenir n échecs. Dit autrement t suit une loi binomiale négative de paramètres n et $1/2$. Le lemme 1.28 utilisant la technique de majoration de Chernoff permet de conclure. □

Remarque : La construction d'une skip list sur un ensemble de n clés peut s'obtenir après tri de ses clés en temps proportionnel à sa taille, i.e. en temps $O(n \log n + t) = \tilde{O}(n \log n)$.

Soit K une clé, fixée une fois pour toute, à rechercher dans une skip list. On note K_i la plus grande clé du niveau i (i.e. de M_i) inférieure ou égale à K. On note également X_i le nombre de clés comprises, au sens large, entre K_{i+1} et K_i dans M_i. Lorsque M_i est vide, on pose $X_i = 0$ et $K_i =$ l'élément fictif minimal (voir plus haut). Pour rechercher K dans la skip list on commence par parcourir les clés du plus haut niveau M_r dans l'ordre croissant jusqu'à atteindre K_r. À l'aide du pointeur de K_r vers le niveau $r-1$ on descend sur sa copie dans M_{r-1} puis on parcourt M_{r-1} dans l'ordre croissant des clés jusqu'à atteindre K_{r-1}. La procédure se poursuit récursivement jusqu'au premier niveau. La recherche est fructueuse si et seulement si $K_1 = K$.

Lemme 4.5 *Le temps de recherche d'une clé fixée est un $\tilde{O}(\log n)$ et le temps moyen est un $O(\log n)$.*

Preuve : Le temps de recherche est clairement proportionnel à la longueur ℓ du "chemin" de recherche, i.e. à $\ell = \sum_{i \geq 1} X_i$ (cette longueur inclut les marches horizontales et verticales).

Notons que pour M_i fixé, M_{i+1} est obtenu en tirant aléatoirement et indépendamment chaque clé de M_i avec une probabilité $1/2$. Par conséquent, la variable aléatoire $X_i|M_i$ suit une loi géométrique de paramètre $1/2$ (plus précisément $X_i|M_i$ est majorée par une variable suivant une telle loi). On en déduit $E(X_i|M_i) \leq 2$, d'où inconditionnellement $E(X_i) \leq 2$.

On note Y_i la variable aléatoire valant 0 si M_i est vide et 1 sinon, de sorte que $X_i \leq nY_i$. On a $P(Y_i = 1) = P(h \geq i) \leq n/2^{i-1}$, soit $E(Y_i) \leq n/2^{i-1}$. On obtient

$$E(\sum_{i\geq 1} X_i) = \sum_{i\geq 1} E(X_i) \leq \sum_{i=1}^{c\log n} E(X_i) + \sum_{i>c\log n} E(nY_i) \leq 2c\log n + 2/n^{c-2}.$$

Et on conclut $E(\ell) = O(\log n)$.

Montrons que $\ell = \tilde{O}(\log n)$. Pour cela on 'coupe' ℓ en deux en écrivant $\ell = \ell_\leq + \ell_>$ avec $\ell_\leq = \sum_{i=1}^{c\log n} E(X_i)$ et $\ell_> = \sum_{i>c\log n} E(X_i)$. Alors $\ell_\leq$ est une somme de $c\log n$ variables aléatoires majorées par des variables de loi géométrique de paramètre $1/2$ et est donc majorée par une variable aléatoire suivant une loi binomiale négative de paramètres $c\log n$ et $1/2$. On en déduit par la technique de Chernoff (lemme 1.28) que

$$P(\ell_\leq > c(2+d)\log n) \leq \exp(-dc\log n/4) = O(\frac{1}{n^{dc/4}})$$

dès que $d \geq 3$. D'où $\ell_\leq = \tilde{O}(\log n)$.

Par ailleurs $\ell_>$ est trivialement majorée par la hauteur h de la skip list additionnée au nombre d'éléments restant au niveau $c\log n + 1$. Soit Z_i la fonction indicatrice de la présence de la i-ème clé au niveau $c\log n + 1$, de sorte que $\ell_> \leq h + \sum_{i=1}^n Z_i$. Les Z_i sont indépendantes et suivent une loi de Bernoulli de paramètre $1/2^{c\log n} = 1/n^c$. Donc $\sum_{i=1}^n Z_i$ suit une loi binomiale de paramètre n et $1/n^c$. Par l'inégalité de Markov on a

$$P(\sum_{i=1}^n Z_i > c) \leq E(\sum_{i=1}^n Z_i)/c \leq \frac{1}{cn^{c-1}}.$$

D'où $\sum_{i=1}^n Z_i = \tilde{O}(1)$. D'après le lemme 4.3, $h = \tilde{O}(\log n)$ et on conclut finalement que $\ell_> = \tilde{O}(\log n)$ puis que $\ell = \tilde{O}(\log n)$. □

On a en fait un résultat plus fort ne dépendant pas de la clé particulière à chercher :

Lemme 4.6 *Le temps maximal de recherche d'une clé quelconque dans une skip list est un $\tilde{O}(\log n)$.*

Preuve :

$$P(\max_K \ell(K) > c\log n) \leq \sum_K P(\ell(K) > c\log n) \leq \frac{n}{p(n,c)} = \frac{1}{q(n,c)}.$$

□

Lemme 4.7 *Le temps maximal de suppression d'une clé quelconque dans une skip list est un* $\tilde{O}(\log n)$.

Pour supprimer une clé on effectue sa recherche en la supprimant des niveaux M_i où elle apparaît. Le temps de suppression est donc de l'ordre du temps de recherche. Si on dispose d'un pointeur sur l'élément à supprimer alors la suppression est un $\tilde{O}(1)$ car $h_K = \tilde{O}(1)$.

Lemme 4.8 *Le temps maximal d'insertion d'une clé dans une skip list est un* $\tilde{O}(\log n)$.

Pour insérer un clé on effectue des tirages jusqu'à obtenir un échec. Si k est le nombre de tirages effectués on insère la clé dans les k premiers niveaux en même temps que l'on effectue une recherche dans la skip list.

Référence :

- Skip Lists : A probabilistic Alternative to Balanced Trees. W. Pugh. Communication of the ACM, 33(6), june 1990.

4.2.2 Arbres binaires de recherche aléatoires

Étant donnée une permutation σ de $[1, n]$ on construit un arbre binaire de recherche en introduisant $\sigma(1)$ puis $\sigma(2)$, ..., puis $\sigma(n)$ dans un arbre initialement vide. Les $n!$ permutations de $[1, n]$ permettent d'obtenir toutes les formes possibles d'arbres binaires de recherche sur n éléments. En considérant une loi de probabilité uniforme sur les $n!$ permutations on en déduit une loi de probabilité sur les arbres binaires de recherche.

Proposition 4.9 *La hauteur d'un arbre binaire de recherche aléatoire est un* $\tilde{O}(\log n)$ *et sa profondeur moyenne est un* $O(\log n)$.

Notons que d'après le théorème 1.31, la hauteur moyenne d'un arbre de recherche aléatoire est un $O(\log n)$, ce qui implique évidement que la profondeur moyenne des clés est du même ordre. On donne ici une preuve de ce dernier résultat, moins fort mais plus simple à montrer.

Preuve : On note X_j^i la variable aléatoire indiquant si j est un ancêtre de i dans l'arbre binaire de recherche associé à une permutation aléatoire σ. On note τ la permutation (aléatoire) inverse de σ. $X_j^i = 1$ si et seulement si $\tau(j) = \min_{k\in[i,j]} \tau(k)$, c.a.d. si et seulement si j est inséré avant i et aucun élément inséré avant j ne sépare i et j. Clairement $P(X_j^i = 1 \,|\, \tau([i, j])) = \frac{1}{|i-j|+1}$, d'où $E(X_j^i) = \frac{1}{|i-j|+1}$.

Soit $h_i = \sum_{j\neq i} X_j^i$ la hauteur de i dans l'arbre binaire de recherche. Par linéarité de l'espérance $E(h_i)$ vaut $\sum_{j=1}^{n} \frac{1}{|i-j|+1} = H_i + H_{n+1-i} - 1 = O(\log n)$.

Par ailleurs, pour i fixé et $i < j$ le lemme 4.10 ci-dessous indique que les variables X_j^i sont mutuellement indépendantes. Par la technique de Chernoff on peut donc écrire pour tout t et tout λ positif

$$P(\sum_{j>i} X_j^i > t) \leq \frac{\prod_{j>i} E(\exp(\lambda X_j^i))}{\exp(\lambda t)}.$$

Or

$$E(\exp(\lambda X_j^i)) = \frac{\exp(\lambda)}{j-i+1} + 1 - \frac{1}{j-i+1} = 1 + \frac{\exp(\lambda)-1}{j-i+1} \leq \exp(\frac{\exp(\lambda)-1}{j-i+1})$$

d'où

$$P(\sum_{j>i} X_j^i > t) \leq \exp((\exp(\lambda)-1)H_n - \lambda t) \leq \exp((\exp(\lambda)-1)(\ln n + 1) - \lambda t).$$

En choisissant $t = c \ln n$ et $\lambda = \ln c$ on trouve

$$P(\sum_{j>i} X_j^i > c \ln n) \leq \frac{\exp(c-1)}{n^{c(\ln c - 1)+1}} = \frac{1}{p(n,c)}.$$

En opérant de même avec la somme $\sum_{j<i} X_j^i$ on en déduit que $h_i = \tilde{O}(\log n)$. Comme pour l'analyse de la hauteur d'une skip list on conclut finalement que la hauteur $h = \max_i h_i$ d'un arbre binaire de recherche aléatoire est un $\tilde{O}(\log n)$ et que son espérance est un $O(\log n)$. □

Dit autrement un arbre binaire de recherche aléatoire est presque toujours bien équilibré.

Lemme 4.10 *Avec les notations de la preuve ci-dessus, soient* $1 \leq i < j_1 < j_2 < \ldots < j_k \leq n$. *Alors les variables* $X_{j_1}^i, X_{j_2}^i \ldots, X_{j_k}^i$ *sont mutuellement indépendantes. On a un résultat analogue pour* $1 \leq j_1 < j_2 < \ldots < j_k < i \leq n$.

Preuve : On raisonne par récurrence sur k. Soient $\epsilon_r \in \{0,1\}$, $r \in [1,k]$. On pose $A = \{X_{j_1}^i = \epsilon_1 \wedge \ldots \wedge X_{j_{k-1}}^i = \epsilon_{k-1}\}$. Par hypothèse de récurrence à l'ordre $k-1$, les variables $X_{j_1}^i, X_{j_2}^i \ldots, X_{j_{k-1}}^i$ sont indépendantes. On remarque en particulier que $P(A) = \prod_{r=1}^{k-1} P(X_{j_r}^i = \epsilon_r)$ ne dépend que des grandeurs $|j_r - i|$. On considère une partie $I \subset [1,n]$ à $j_k - i + 1$ éléments et le sous-ensemble de permutations de $[1,n]$:

$$B_I = \{\tau : \tau([i,j_k]) = I \wedge \tau(j_k) = \min I\}$$

On note que les B_I sont disjoints et que leur union, $\cup_I B_I$, est l'événement $\{X_{j_k}^i = 1\}$. J'affirme que $P(A|B_I) = P(A)$. Pour le voir on considère l'injection croissante $\iota : [1, j_k - i] \to [1,n]$ telle que $\mathrm{Im}\iota = I$ et la surjection ϕ de B_I dans les permutations de $[1, j_k - i]$:

$$\begin{array}{ccc} B_I & \stackrel{\phi}{\to} & \mathcal{S}_{j_k - i} \\ \tau & \mapsto & \tau' : \ell \mapsto \iota^{-1}(\tau(\ell + i - 1)) \end{array}$$

Puisque l'appartenance de τ à A ne dépend que des ordres relatifs de $\tau(i), \tau(i+1), \ldots, \tau(j_{k-1})$, on a

$$\tau \in A \Leftrightarrow \phi(\tau) \in A' = \{X_{j_1 - i + 1}^i = \epsilon_1, X_{j_2 - i + 1}^i = \epsilon_2, \ldots, X_{j_{k-1} - i + 1}^i = \epsilon_{k-1}\}$$

De plus, le nombre $|\phi^{-1}(\tau')|$ ne dépend pas de $\tau' \in \mathcal{S}_{j_k - i}$. On en déduit, avec la remarque ci-dessus, $P(A|B_I) = P(A') = P(A)$, et par suite $P(A|X_{j_k}^i = 1) = P(A)$ (cf. exercice 1.20). En reportant cette égalité dans la suivante

$$P(A) = P(A|X_{j_k}^i = 1)P(X_{j_k}^i = 1) + P(A|X_{j_k}^i = 0)P(X_{j_k}^i = 0)$$

on obtient également $P(A|X^i_{j_k} = 0) = P(A)$. Finalement on en déduit,

$$\forall \epsilon_k \in \{0,1\} : P(A \wedge X^i_{j_k} = \epsilon_k) = P(A)P(X^i_{j_k} = \epsilon_k)$$

soit encore

$$P(X^i_{j_1} = \epsilon_1 \wedge \ldots \wedge X^i_{j_k} = \epsilon_k) = \prod_{r=1}^{k} P(X^i_{j_r} = \epsilon_r)$$

□

4.2.3 Tree + Heap = Treap

Introduits sous le nom d'arbres cartésiens par J. Vuillemin, les treaps sont des arbres binaires de recherches dont les noeuds possèdent une clé et une priorité comprise entre 0 et 1. Un treap est un arbre binaire de recherche pour les clés - les noeuds dans le sous arbre gauche (droit) d'un noeud ont une clé inférieure (supérieure) à la clé de ce noeud - et un tas pour les priorités (le parent d'un noeud a une priorité plus grande que celle de ce noeud). Un treap est donc le résultat d'un arbre binaire de recherche construit en introduisant un à un les éléments dans l'ordre de leur priorité. En choisissant les priorités de manière aléatoire on peut simuler un arbre binaire de recherche aléatoire. Ceci assure d'après le lemme 4.9 que la recherche a un coût en $\tilde{O}(\log n)$. Il est possible de maintenir dynamiquement un treap, i.e. d'insérer ou de supprimer des éléments en conservant la propriété d'arbre aléatoire. Pour cela, on utilise le fait que l'opération de rotation sur un arbre binaire (cf. section 1.4) préserve l'ordre infixe, intervertit la parenté pour deux noeuds, et préserve la propriété de tas pour le reste. Ceci permet, par rotations successives de faire remonter ou descendre un noeud dans un treap pour l'ôter ou le positionner à sa bonne place, c'est-à dire comme s'il avait été introduit 'à l'instant' donné par sa priorité.

Références :

- Randomized search trees. C. Aragon and R. Seidel. Algorithmica 16. pp 464-497. 1996.

- Randomized Binary Search Trees. C. Martínez and S. Roura. Journal of the ACM. 45. pp 288-323. 1998.

Chapitre 5

Arrangements

5.1 Introduction : problème de la discrépance

On se donne un ensemble fini $\mathcal{P}$ de points dans le carré unité et on compare la mesure de l'intersection d'un demi-plan avec le carré avec la proportion de points de $\mathcal{P}$ contenu dans cette intersection. On cherche le maximum de la différence entre ces valeurs pour tous les demi-plans possibles.

Le maximum est obtenu avec des droites qui passent soit par au moins deux points de $\mathcal{P}$ soit par un seul sommet mais pour des positions extrémales, dues à la forme du carré.

On traite explicitement ces derniers cas en temps $O(n^2)$. Pour le cas général les droites passant par deux sommets apparaissent comme des sommets de l'arrangement des droites duales de $\mathcal{P}$. Il reste à voir combien de sommets sont de part et d'autres de ces droites passant par deux sommets. Pour cela on construit explicitement la carte de l'arrangement dual et on marche le long de chaque droite à partir d'une extrémité. Il est facile de maintenir pour chaque sommet rencontré au cours de cette marche son *niveau*, i.e. le nombre de droites duales strictement au dessus de ce sommet.

5.2 Préliminaire : subdivision du plan

Une *k-cellule* du plan est un sous-espace du plan homéomorphe à une k-boule ouverte. Une *subdivision du plan* est un ensemble fini de cellules disjointes de dimension 0, 1 et 2 du plan dont l'union est le plan de sorte que la frontière d'une cellule est une union de cellules de dimensions inférieures. Dans la pratique on considère souvent que le plan est compactifié par l'ajout d'un point "à l'infini" de sorte que la subdivision porte véritablement sur la sphère et non sur le plan. Pour cette section on pourra également consulter le chapitre 2 de ces notes.

Un graphe est planaire s'il peut être plongé dans le plan. Un graphe plan – ou carte planaire – est un graphe plongé dans le plan. Tout graphe plan connexe induit une subdivision du plan dont les *faces* sont les composantes bornées du complémentaire du graphe.

Théorème 5.1 (Relation d'Euler) *Tout graphe plan connexe ayant s sommets, a arêtes et f faces vérifie : $s - a + f = 1$.*

Preuve : Par récurrence sur le nombre d'arêtes. Trivial si $a = 1$. Si le graphe contient un cycle, alors on ôte une arête de ce cycle qui borde deux faces distinctes (par le théorème de séparation Jordan). L'hypothèse de récurrence s'applique donc avec une face et une arête en moins. Sinon le graphe est sans cycle et contient un sommet de degré un. En ôtant ce sommet et l'arête incidente on peut à nouveau appliquer l'hypothèse de récurrence avec un sommet et une arête en moins. □

Note : généralement on considère la relation d'Euler pour un graphe tracé sur la sphère. On passe du plan à la sphère via une projection stéréographique. La face non-bornée du plan devient une face pour la sphère d'où la relation d'Euler pour la sphère : $s-a+f = 2$.

Soit $\pi : G \to \mathbb{R}^2$ un plongement polygonal d'un graphe planaire G (les plongements des arêtes sont donc des arcs polygonaux). On définit pour chaque sommet s du graphe un ordre circulaire sur les arêtes $a_1, \ldots, a_k$ incidentes à ce sommet par l'ordre circulaire dans le sens indirecte autour de $\pi(s)$ des segments de $\pi(a_1), \ldots, \pi(a_k)$ incidents à $\pi(s)$. Si le plongement n'est pas polygonal on peut encore définir un ordre circulaire autour de chaque sommet via un homéomorphisme du plan qui transforme les arcs intérieurs à un petit disque centré en $\pi(s)$ en des arcs polygonaux tout en laissant fixe l'extérieur du disque. On montre [MT01, p. 88] que cet ordre est indépendant de l'homéomorphisme employé. La donnée *combinatoire* du graphe G avec ces ordres circulaires est appelé une *carte combinatoire* ou *système de rotations*.

On montre [MT01, cor. 3.2.5] qu'une carte combinatoire détermine un unique plongement de G à homéomorphisme près. Dit autrement, si $\pi_1, \pi_2 : G \to \mathbb{R}^2$ sont deux plongements de G qui induisent la même carte combinatoire alors il existe un homéomorphisme ϕ du plan tel que $\pi_2 = \phi \circ \pi_1$. Dans la pratique on représente souvent une carte combinatoire à l'aide d'une liste de *demi-arêtes* représentant chacune une des deux orientations possibles de chaque arête du graphe. À chaque demi-arête a on adjoint deux pointeurs *inv* et *rot* respectivement vers la demi-arête d'orientation opposée et vers la demi-arête suivant a dans le sens indirecte autour du sommet origine de a. Notons que la demi-arête suivant a dans la face située à gauche de a (en regardant de l'origine vers la pointe de la demi-arête) s'obtient en suivant le pointeur *inv* puis *rot*. La structure porte souvent le nom de DCEL pour Doubly-Connected Edge List. Une telle structure peut être définie pour un plongement dans la sphère et plus généralement dans une surface orientable de genre quelconque.

Références :

- [MT01]
- Primitives for the manipulation of three-dimensional subdivisions. L. Guibas and J. Stolfi, ACM Transactions on Graphics, 1985, 4(2), pp 74-123.

5.3 Arrangement de droites

Définition 5.2 *Un* arrangement *d'un ensemble de droites du plan est la décomposition du plan induite par ces droites. Les 2-faces ou cellules de cet arrangement sont les composantes connexes du complémentaire des droites, les 1-faces et 0-faces sont les faces des fermetures de ces cellules. Un arrangement est* simple *si deux droites se coupent toujours mais trois droites ont une intersection vide.*

Théorème 5.3 *Un arrangement de n droites a au plus $\binom{n}{2}$ sommets, n^2 arêtes et $\binom{n}{2} + n + 1$ faces.*

Preuve : Ces bornes sont exactes pour un arrangement simple : chaque paire de droites s'intersecte en un sommet et chaque droite est coupée en n segments par les $n-1$ autres droites. Par la relation d'Euler, obtenue sur la sphère en ajoutant un sommet à l'infini au plan, on obtient le nombre de faces. On peut se passer de la relation d'Euler en utilisant un balayage par une droite non parallèle à celle de l'arrangement. Le nombre de faces à gauche de cette droite passe de 0, à l'extrême gauche, au nombre de sommets, une fois à droite du dernier sommet de l'arrangement, auquel on doit ajouter $n+1$ faces à l'extrême droite. Ces égalités deviennent des majorations dans le cas d'un arrangement quelconque. En effet, en perturbant de manière infinitésimale un arrangement non simple $\mathcal{A}$ on obtient un arrangement simple $\mathcal{A}'$ de sorte que chaque sommet de degré d de $\mathcal{A}$ soit éclaté en un arrangement simple de $d/2$ droites. Pour cela, il suffit de déplacer chaque droite une à une dans un voisinage tubulaire, ne contenant aucun autre sommet que ceux qu'elle contient, de sorte que tous les sommets sur cette droite soient simples et que cette droite ne soit parallèle à aucune autre. Chaque droite avant perturbation étant coupée au plus $n-1$ fois, le nombre de segments de $\mathcal{A}'$ majore celui de $\mathcal{A}$. De même pour le nombre de sommets, puisque chaque paire de droites définie au plus un sommet. Pour obtenir une majoration du nombre de facettes on peut associer à chaque facette de $\mathcal{A}$ un point intérieur. Ces points restent intérieurs au complémentaire des droites après perturbation (suffisamment petite pour cela) et restent dans des composantes connexes distinctes car tout segment joignant deux de ces points coupe une droite de l'arrangement avant et après perturbation. □

Définition 5.4 *La* zone *d'une droite dans un arrangement est l'union des faces de l'arrangement dont la fermeture intersecte cette droite, et des bords de ces faces. La* complexité *de cette zone est le nombre total de faces, arêtes et sommets qu'elle contient.*

Théorème 5.5 (de la zone) *Le nombre d'arêtes de la zone d'une droite dans un arrangement de n droites du plan est majoré par $6n$.*

Preuve 1 : Soient $L = \{l_1, l_2, \ldots, l_n\}$ les n droites d'un arrangement et l une autre droite dont la direction est dite horizontale (pente nulle). On considère l'ensemble E des arêtes de la zone $Z(\ell)$ de ℓ qui sont strictement au dessus de ℓ (donc non incidente à ℓ). On associe à toute arête e non-horizontale (resp. horizontale et bornée à gauche) de E la droite $\phi(e) \in L$ qui suit la droite support $\ell(e)$ de e en tournant dans le sens indirect

autour de l'extrémité inférieure (resp. extrémité gauche) de e. Notons que $\phi(e)$ ne peut être horizontale. Montrons que pour toute $\ell_i \in L$ on a $|\phi^{-1}(\ell_i)| \leq 2$. Dans le contraire on note e et e' les deux arêtes de $\phi^{-1}(\ell_i)$ (dont les extrémités sur ℓ_i sont) les plus basses. On vérifie que les deux dièdres "supérieurs" définis par ℓ_i et $\ell(e)$ d'une part et par ℓ_i et $\ell(e')$ d'autre part sont de part et d'autre de ℓ_i et ne coupent pas ℓ. Les autres arêtes de $\phi^{-1}(\ell_i)$ sont contenues dans l'intérieur de l'union de ces dièdres et ne sont donc pas incidentes à une face de $Z(\ell)$, une contradiction. On déduit de ce qui précède $|E| \leq 2n$. (S'il y a une droite horizontale alors $|E| \leq 2(n-1)+1$).

Le nombre d'arêtes supérieures ou inférieures de la zone et non-incidentes à ℓ est donc majoré par $4n$. L'ensemble des arêtes de $Z(\ell)$ incidentes à ℓ est par ailleurs majoré par $2n$ puisque chaque droite de L donne naissance à au plus deux telles arêtes. □

La preuve qui suit permet de majorer par $8n-4$ le nombre de couples (e, f) où e est une arête incidente à une face f de la zone.

Preuve 2 : Avec les notations de la preuve 1, on oriente les l_i non horizontales vers le bas et les autres vers la droite. On considère ensuite une droite l' infinitésimalement translatée au dessus de l. Cette droite coupe les faces dont la fermeture intersecte l et situées (partiellement) au dessus de l selon une suite ordonnée de gauche à droite. Pour chacune de ces faces on écrit la chaîne des droites supports de ses arêtes situées au dessus de l dans un parcours dans le sens indirecte du bord de la face, en commençant par une arête incidente à l. On ajoute par ailleurs un signe + (resp. -) à chaque droite support selon son sens de parcours. La concaténation de gauche à droite de ces listes fournit un mot sur les symboles $\pm l_i$. On scinde ce mot en deux sous-mots L_+ et L_- selon les signes des symboles.

Remarque : Une arête a non horizontale (resp. horizontale) de L_+ a sa face $f(a)$ à gauche (resp. en dessous) de sa droite support $l(a)$ en raison du sens de parcours des faces. Par conséquent toutes les faces à gauche de $f(a)$ (relativement à l') sont également à gauche de $l(a)$.

Les sous-mots L_+ et L_- vérifient les propriétés suivantes :
- ils ne contiennent pas de facteur $l_i l_i$ car deux arêtes portées par les mêmes droites appartiennent à des faces distinctes séparées par un chemin joignant l_i à l. Le dernier segment de ce chemin est porté par une droite $l_j \neq l_i$ et est parcouru dans les deux sens. Autrement dit pour toute double occurrence de l_i dans L_+ ou L_- il existe un l_j qui les sépare.
- ils ne contiennent pas de sous mot de la forme $l_i l_j l_i l_j$. Supposons que $l_i l_j l_i$ est sous-mot de L_+. Soient a_g, b et a_d les segments correspondant respectifs. On forme un cycle avec la chaîne gauche c_g (resp. droite c_d) de $f(a_g)$ (resp. $f(a_d)$) comprise entre l et l_i et les segments des droites l et l_i joignant $f(a_g)$ et $f(a_d)$. Ce cycle est convexe donc simple. b rencontre l'intérieur de cette région de part sa position dans L_+. l_j ne rencontrant aucune des deux chaînes c_g et c_d rencontre l_i entre a_g et a_d (ainsi que l (Jordan again)). On en déduit que a_d est à droite de l_j (qui ne peut être horizontale). La remarque précédente montre qu'une nouvelle occurrence de l_j dans L_+ impliquerait que a_d est à gauche de l_j, une contradiction.

De tels mots sont appelés (n,2)-suites de Davenport-Shinzel. Ce sont précisément des mots formés sur une alphabet à n lettres, ne comportant pas de facteur aa ni de sous-mot $abab$. Le lemme suivant permet de conclure. □

Lemme 5.6 *Une (n,2)-suite de Davenport-Shinzel est de longueur au plus* $2n-1$.

Preuve : Soit S une (n,2)-suite de Davenport-Shinzel. Le lemme est trivial si chaque lettre n'apparaît qu'une fois. Sinon on considère le plus petit facteur S' de S séparé par deux occurrences d'un même lettre de sorte que $S = S_1 a S' a S_2$. En posant $k = |S'|$, on vérifie que $S_1 a S_2$ est une (n-k,2) suite de Davenport-Shinzel. On conclut par récurrence sur n que $|S| \leq 2(n-k) - 1 + k + 1 \leq 2n - 1$. □

Corollaire 5.7 *La complexité de la zone d'une droite dans un arrangement de n droites du plan est linéaire (i.e. en* $O(n)$*).*

Preuve : Le théorème précédent borne le nombre d'arêtes de la zone. Le nombre de faces est par ailleurs majoré par $2n$ et le nombre de sommets est borné par le nombre d'arêtes de la zone. □

On construit incrémentalement la carte d'un arrangement en introduisant chaque droite une à une dans la carte déjà construite. Initialement, on introduit une première droite en ajoutant un sommet à l'infini p_∞ à la carte qui représente donc une sphère (le plan compactifié). Chaque droite introduite est donc doublement incidente à ce sommet. Pour introduire une nouvelle droite ℓ, on l'oriente et on part de la face ayant deux arêtes incidentes à p_∞ dont les deux vecteurs directeurs (issus de p_∞) encadrent celui de ℓ. On marche dans cette face jusqu'à trouver la sortie – une arête ou un sommet coupé par ℓ – et on modifie la carte en conséquence. On poursuit avec la face suivante : si on sort par l'intérieur d'une arête alors la face suivante est bien définie. Sinon il faut tourner autour du sommet de sortie pour la trouver. Voir la figure 5.1. La complexité d'insertion d'une

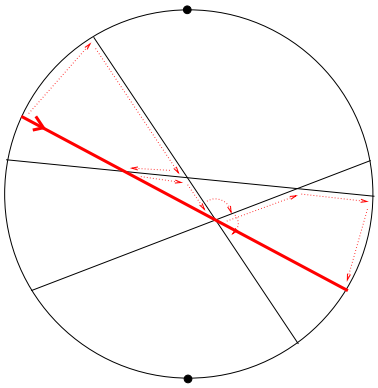


FIG. 5.1 – La droite épaisse est introduite en partant de son point infini à gauche. Le cercle représente un éclatement du point à l'infini.

droite est proportionnelle à la complexité de sa zone dans l'arrangement déjà construit. Compte tenu du théorème 5.5 on en déduit :

Proposition 5.8 *L'arrangement de n droites dans le plan peut être calculé en temps optimal $O(n^2)$.*

5.4 Arrangement d'hyperplans

Soit H un ensemble d'hyperplans dans E^d.

Définition 5.9 *L'arrangement $\mathcal{A}(H)$ de H est la subdivision de E^d en polyèdres convexes induite par H.*

Définition 5.10 *un* k-flat *est un sous-espace affine de dimension k. C'est encore l'enveloppe affine de $k+1$ points affinement indépendants. Il est* vertical *s'il contient la direction verticale.*

Si un plan est non vertical on peut donc parler des demi-espaces *dessous* et *dessus* cet hyperplan.

Étant donné un arrangement d'hyperplans non verticaux $H = \{h_i\}_{1 \leq i \leq n}$, on associe à chaque point p de E^d un *vecteur position* dans $\{-1, 0, 1\}^n$ dont la $i$ème composante indique si p est dessous, sur ou dessus h_i.

On définit les *faces* de $\mathcal{A}(H)$ comme les composantes connexes du complémentaire de $\cup_i h_i$ dans $\mathbb{R}^d$ auxquelles on ajoute récursivement les faces des arrangements induits dans chaque h_i par les intersections de h_i avec les hyperplans de $H \setminus \{h_i\}$. On vérifie que deux points d'une même face de $\mathcal{A}(H)$ ont même vecteur position et que ce vecteur position définit de manière univoque cette face. Notons cependant que tout vecteur position de $\{-1, 0, 1\}^n$ ne correspond pas nécessairement à une face de l'arrangement. Le vecteur position d'une sous-face d'une face est obtenu en annulant certaines des composantes du vecteur position de la face. Une *k-face* est une face de dimension k, c'est-à-dire dont l'enveloppe affine est un k-flat. Ainsi, une 0-face est un sommet et une 1-face est une arête. Une d (resp. $(d-1)$)-face est encore appelée *cellule* (resp. *facette*).

Un arrangement est *simple* si l'intersection de d hyperplans quelconques de l'arrangement est toujours réduite à un point tandis que l'intersection de $d+1$ hyperplans est toujours vide. S'il y a strictement moins de d hyperplans on demande que leur intersection soit un $(d-n)$-flat. De manière équivalente un arrangement est simple si toute face de codimension k est incluse dans exactement k hyperplans, i.e. a exactement k zéros dans son vecteur position et si les directions de toute famille d'au plus d hyperplans forment une famille libre. En particulier une sous-face d'une face a exactement un zéro de plus dans son vecteur position.

5.4.1 Dénombrement des faces et incidences

On note $f_k(H)$ le nombre de k-faces de l'arrangement H et $i_k(H)$ le nombre d'incidences entre les k-faces et les $(k+1)$-faces de H. On note également $f_k^d(n)$ et $i_k^d(n)$ le maximum de ces nombres sur tous les arrangements de n hyperplans dans E^d.

Lemme 5.11 *Pour tout arrangement simple H de $n \leq d$ hyperplans dans E^d on a :*

$$f_k(H) = \binom{n}{d-k} 2^{n-d+k} \text{ et } i_k(H) = 2(d-k)f_k(H)$$

De plus chaque cellule de H est incidente à $\binom{n}{d-k}$ k-faces.

Preuve : Utiliser le vecteur position et le fait que chaque vecteur position correspond effectivement à une face. On peut en effet écrire l'équation du $i$ème hyperplan sous la forme $x_i = 0$ dans une base convenablement choisie. □

Lemme 5.12 *Pour tout arrangement simple H de n hyperplans dans E^d on a*

$$f_k(H) = \sum_{i=0}^{k} \binom{d-i}{k-i}\binom{n}{d-i} = \binom{n}{d-k}\sum_{i=0}^{k}\binom{n-d+k}{k-i} \text{ et } i_k(H) = 2(d-k)f_k(H).$$

En particulier, pour d constant on a $f_k(H) = \Theta(n^d)$ et $i_k(H) = \Theta(n^d)$.

Preuve : Pour $n \leq d$ c'est le lemme 5.11. La preuve précédente pour le nombre d'incidences est encore valide pour $n > d$. On fait une preuve par récurrence sur d pour $f_k(H)$ lorsque $n > d$: le lemme est trivialement vrai en dimension 1. On suppose qu'il est vrai en dimension $d-1$. Soit H comme dans l'énoncé du lemme. Puisque l'arrangement est simple il a $f_0(H) = \binom{n}{d}$ sommets. On choisit une direction d'hyperplan ne contenant aucune des droites vectorielles déterminées par les paires de sommets de l'arrangement de H. On balaye cet hyperplan de gauche à droite. À chaque fois que l'on balaye un sommet p de l'arrangement une cellule de plus se trouve à gauche de l'hyperplan de balayage. Cette cellule c correspond à l'une des cellules du sous-arrangement simple des d hyperplans de H s'intersectant au point p. Les k-faces supplémentaires balayés en p sont toutes des faces de c et sont donc au nombre de $\binom{d}{d-k}$ (utiliser le vecteur position comme pour la preuve du lemme 5.11). Lorsqu'on balaye le sommet le plus à droite on a donc balayé un nombre $f_0(H)\binom{d}{d-k}$ de k-faces. Il reste à compter les k-faces à l'extrême droite de l'arrangement. Elles sont en bijection avec les $(k-1)$-faces de l'arrangement simple de dimension $d-1$ obtenu en coupant $\mathcal{A}(H)$ par un hyperplan de balayage situé à droite du dernier sommet. Par hypothèse de récurrence il y a $\sum_{i=0}^{k-1}\binom{d-1-i}{k-1-i}\binom{n}{d-1-i}$ telles faces. D'où :

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

□

Théorème 5.13 *Les valeurs de $f_k(H)$ et $i_k(H)$ du lemme 5.12 dans le cas simple sont des bornes supérieures dans le cas général, i.e. :*

$$f_k^d(n) = \sum_{i=0}^{k}\binom{d-i}{k-i}\binom{n}{d-i} \text{ et } i_k^d(n) = 2(d-k)f_k^d(n).$$

En considérant d constant on a en particulier

$$f_k^d(n) = \Theta(n^d) \text{ et } i_k^d(n) = \Theta(n^d).$$

Preuve : On utilise une double récurrence sur n et d. Le théorème est trivialement vrai pour $d = 1$ et n quelconque et pour $n = 2$ et d quelconque. On suppose le théorème vrai pour n hyperplans jusqu'à la dimension $d-1$ et en toute dimension pour au plus $n-1$ hyperplans. On se donne une famille $H = \{h_i\}_{1 \leq i \leq n}$ de n hyperplans dans E^d et on pose $G = H \setminus \{h_n\}$. L'hyperplan h_n intersecte $\mathcal{A}(G)$ en un arrangement $\mathcal{A}(G')$ de dimension $d-1$ de $n-1$ hyperplans au plus. Une k-face de $\mathcal{A}(H)$ est soit
- une k-face de $\mathcal{A}(G)$,
- une k-face de $\mathcal{A}(G')$ ou
- une k-face de $\mathcal{A}(G)$ coupée en deux par h_n et qui correspond donc à une $(k-1)$-face de $\mathcal{A}(G')$.

On en déduit

$$f_k(H) \leq f_k(G) + f_k(G') + f_{k-1}(G').$$

Par hypothèse de récurrence on en déduit

$$\begin{aligned} f_k(H) &\leq \sum_{i=0}^{k} \binom{d-i}{k-i}\binom{n-1}{d-i} + \sum_{i=0}^{k} \binom{d-1-i}{k-i}\binom{n-1}{d-1} + \sum_{i=0}^{k-1} \binom{d-1-i}{k-1-i}\binom{n-1}{d-1-i} \\ &= \sum_{i=0}^{k} \binom{d-i}{k-i}\binom{n}{d-i}. \end{aligned}$$

Une paire de faces incidentes de dimensions respectives k et $k+1$ de $\mathcal{A}(H)$ provient soit
- d'une telle paire dans $\mathcal{A}(G)$,
- d'une telle paire dans $\mathcal{A}(G')$,
- d'une $(k+1)$-face de $\mathcal{A}(G)$ coupée en deux par h_n et de la k-face correspondant à leur intersection ou
- d'une paire dans $\mathcal{A}(G)$ coupée en deux par h_n et qui correspond donc a une paire de faces incidentes de dimension k et $k-1$ dans $\mathcal{A}(G')$.

On en déduit

$$i_k(H) \leq i_k(G)+i_k(G')+2f_k(G')+i_{k-1}(G') \leq 2(d-k)(f_k(G)+f_k(G')+f_{k-1}(G')) \leq 2(d-k)f_k^d(n).$$

□

Définition 5.14 *Soit H un arrangement de n hyperplans dans E^d et h un $(n+1)$-ième hyperplan. La* zone *de h dans H est l'union des d-faces dont l'adhérence intersecte h et de leurs faces. On note $Z_k(h, H)$ l'ensemble des paires (f, c) où f est une face de codimension k de la d-face c de la zone. On note également $z_k(h, H)$ le cardinal de $Z_k(h, H)$ et $z_k(n, d)$ le maximum de cette valeur pour tout h et H.*

Théorème 5.15 (Théorème de la zone) *Pour d fixé, $z_k(n, d) = O(n^{d-1})$.*

Preuve : On suppose que $H \cup \{h\}$ est un arrangement simple et que $n > d$ (le résultat est asymptotique). On considère un hyperplan g de H et on compare la zone de h dans H avec celle de h dans $H \setminus \{g\}$ et de $h \cap g$ dans H/g (= arrangement de dimension $d-1$ de $H \setminus \{g\}$ intersecté avec g) afin d'établir une formule de récurrence. Plus précisément

on s'intéresse au nombre $n_k(g)$ de paires (f,c) de $Z_k(h,H)$ où f n'est pas contenue dans g. Si l'adhérence $\bar{c}$ de c ne rencontre pas g, alors (f,c) est une paire de $Z_k(h,H\setminus\{g\})$. Sinon (f,c) provient d'une paire (ϕ,κ) de $Z_k(h,H\setminus\{g\})$ coupée en deux par g. On note (f',c') la seconde "moitié" de (ϕ,κ). Si (f',c') est également dans $Z_k(h,H)$ (i.e. si g ne sépare pas f' de h) alors la paire $(\phi\cap g,\kappa\cap g)$ est dans $Z_k(h\cap g,H/g)$ ce qui permet de comptabiliser de manière univoque (f,c) dans $Z_k(h,H\setminus\{g\})$ et (f',c') dans $Z_k(h,H/g)$. On en déduit :

$$n_k(g)\leq z_k(h,H\setminus\{g\})+z_k(h\cap g,H/g).$$

En sommant cette relation sur tous les hyperplans g de H et en remarquant que chaque paire de $Z_k(h,H)$ est comptée $(n-k)$ fois dans cette somme (par hypothèse de simplicité f est contenue dans exactement k hyperplans), on obtient

$$(n-k)z_k(h,H)\leq\sum_{g\in H}z_k(h,H\setminus\{g\})+z_k(h\cap g,H/g).$$

D'où

$$z_k(n,d)\leq\frac{n}{n-k}(z_k(n-1,d)+z_k(n-1,d-1)).$$

Pour résoudre la récurrence, on commence par poser :

$$z_k(n,d)=\binom{n}{k}w_k(n,d)=O(n^kw_k(n,d))$$

pour se ramener à la récurrence :

$$w_k(n,d)=w_k(n-1,d)+w_k(n-1,d-1)$$

Il s'agit alors de montrer que $w_k(n,d)=O(n^{d-1-k})$. On résout cette récurrence en partant du cas planaire $d=2$ déjà traité ($z_k(n,2)=O(n)\implies w_k(n,2)=O(n^{1-k})$). Par hypothèse de récurrence (sur d) on a donc $z_k(n,d-1)=O(n^{d-2})$. On en déduit

$$w_k(n,d)=w_k(n-1,d)+O(n^{d-2-k})=w_k(d,d)+O(\sum_{i=d+1}^{n}i^{d-2-k})$$

Si $d-2-k\geq 0$ on obtient $w_k(n,d)=O(n^{d-1-k})$ comme annoncé. Sinon, pour $k=d-1$ et d, i.e. pour les sommets et les arêtes, on utilise un autre argument : On note $f(0,3,c)$ le nombre de triplets (f_0,f_3,c) où f_0 est un sommet d'une 3-face f_3 d'une cellule c de la zone. On note de même $f(2,3,c)$ le nombre de triplets (f_2,f_3,c) où f_2 est une 2-face d'une 3-face f_3 d'une cellule c de la zone. Comme chaque f_3 est un 3-polyèdre, on a dans une f_3 donnée : $\#f_0\leq 2\#f_2$ (se déduit de la relation d'Euler et du fait que chaque sommet est incident à 3 arêtes au moins), d'où $f(0,3,c)\leq 2f(2,3,c)$. Par ailleurs, l'arrangement étant simple, chaque 2-face de c est contenue dans $(d-2)$ des 3-faces de c (penser au vecteur position). En sommant sur la zone, on en désuit : $\sum_c f(2,3,c)\leq(d-2)f_{d-2}(h,H)$. On en déduit

$$f_d(h,H)\leq\sum_c f(0,3,c)\leq 2\sum_c f(2,3,c)\leq 2(d-2)f_{d-2}(h,H),$$

d'où $f_d(h,H)=O(n^{d-1})$.

Enfin, par simplicité de l'arrangement, la clôture de chaque cellule est un polytope simple et chaque sommet y est incident à d arêtes. D'où $2z_{d-1}(h,H) = dz_d(h,H)$. On conclut avec l'inégalité précédente.

Dans le cas d'un arrangement non simple on peut montrer en perturbant les hyperplans que la complexité de la zone (les $z_k(n,d)$) est majorée par celle d'une configuration simple d'hyperplans. □

On retrouvera la preuve du théorème de la zone selon les mêmes lignes dans [Mul94, Sec. 6.2] et [Mat02, Sec. 6.4].

5.5 Dualité

À un point $x = (\mathbf{x}, x_d) \in E^d$ on associe l'hyperplan $\mathcal{D}(x)$ d'équation $y_d = 2\mathbf{x}.\mathbf{y} - x_d$. On définit ainsi une bijection de E^d dans l'espace des hyperplans non verticaux de E^d. On utilise la même notation pour l'inverse de $\mathcal{D}$.

On note U le paraboloide de révolution d'équation $x_d = \mathbf{x}^2$. Ce qui s'écrit encore $(x_d + \frac{1}{4})^2 = (x_d - \frac{1}{4})^2 + \mathbf{x}^2$. U a donc le point $(\mathbf{0}, \frac{1}{4})$ pour foyer et l'hyperplan $\{x_d = -\frac{1}{4}\}$ pour directrice. On note $T_U(x)$ l'hyperplan tangent en un point x de U.

Propriétés géométriques :

- deux points, p et q sont sur une même verticale si et seulement si $\mathcal{D}(p)$ et $\mathcal{D}(q)$ sont parallèles,
- plus spécifiquement, si τ est une translation verticale alors $\mathcal{D}(\tau.x) = \tau^{-1}.\mathcal{D}(x)$,
- Si $x \in U$ alors $\mathcal{D}(x) = T_U(x)$, ce qui avec la propriété précédente montre que pour x quelconque $\mathcal{D}(x)$ s'obtient comme l'hyperplan symétrique par rapport à $T_U(x')$ de l'hyperplan parallèle à $T_U(x')$ et passant par x, où x' est le point de U à la verticale de x,
- pour x en dessous de U, l'hyperplan tangent à U aux points de $\mathcal{D}(x) \cap U$ passe par x,
- pour x en dessous de U, $\mathcal{D}(x) \cap U$ se projette verticalement sur l'hyperplan horizontal en une (d-1)-sphère dont le centre est la projection de x et le carré du rayon la distance de x au point de U situé à la verticale de x.

Propriétés de dualité :

- $\mathcal{D} \circ \mathcal{D}$ est l'identité (par définition),
- x est dessous (resp. sur, resp. dessus) l'hyperplan non vertical h si et seulement si $\mathcal{D}(h)$ est dessous (resp. sur, resp. dessus) $\mathcal{D}(x)$,
- Soit h un hyperplan non vertical et $Q = \{x_1, \ldots, x_d\}$ un ensemble de d points de h affinement indépendants alors $\mathcal{D}(Q)$ définit un arrangement simple i.e. a un unique point d'intersection.
  Preuve : Un point appartient à l'intersection des hyperplans de $\mathcal{D}(Q)$ si et seulement

si son hyperplan dual contient Q, i.e. si et seulement si son dual est h.

- Soit h un hyperplan non vertical et Q un ensemble quelconque de points dans h. L'enveloppe affine de Q est de dimension k si et seulement si $\mathcal{D}(Q)$ s'intersecte en un sous-espace de dimension $d-k-1$.
  Preuve : Soit Q' une famille de $k+1$ points de Q affinements indépendants. On les complètes en une famille indépendantes de d points de h. D'après le point précédent les hyperplans duaux de cette famille forment un arrangement simple et c'est donc le cas pour la sous famille $\mathcal{D}(Q')$. Leur intersection est donc un sous-espace affine de dimension $d-k-1$. Les autres points de Q étant affinement liés à Q' l'équation de leurs hyperplans duaux sont des combinaisons linéaires de celles de Q' et contiennent donc leur intersection.

- Comme corollaire du point précédent on a :
  Soit Q un ensemble de n points dans E^d et $H = \mathcal{D}(Q)$. Soit h un hyperplan non vertical. L'enveloppe affine des points de Q situés sur h est de dimension k si et seulement si $\mathcal{D}(h)$ est contenu dans l'intérieur d'une $(d-1-k)$-face de $\mathcal{A}(H)$.
  Preuve : les hyperplans duaux des points de Q sur h définissent un sous-espace affine de dimension $d-k-1$. Or les autres points de Q n'étant pas sur h, leur dual ne contient pas $\mathcal{D}(h)$ et évitent une petite boule centrée en ce point. L'intersection de cette boule avec le sous-espace s'étend en une unique $(d-1-k)$-face de $\mathcal{A}(H)$.

- Puisque la dualité préserve les positions relatives (dessous/dessus) entre points et hyperplans non verticaux, un hyperplan dual d'un point p intérieur à une cellule de $\mathcal{A}(H)$ sépare les points de Q en deux ensembles indépendants du point p. La réciproque est cependant fausse car il peut y avoir des cellules dont les vecteurs positions sont "opposés". Cela ne peut arriver que pour des paires de cellules non bornées.

Exercice 5.16 *Donner un algorithme de complexité $O(n \log n)$ pour trouver une boîte rectangulaire parallèle aux axes contenant tous les sommets d'un arrangement de n droites.*

Exercice 5.17 *Soient n points du plan. Donner un algorithme de complexité quadratique pour trouver une droite passant par le plus possible de ces points.*

Exercice 5.18 *Soient n segments du plan. Trouver un algorithme quadratique pour décider si ces segments peuvent être tous percés par une même droite.*

Exercice 5.19 *Étant donnés n points rouges et m points bleus en position générale dans le plan, trouver une droite qui "coupe" simultanément les points rouges et les points bleus en deux, c'est à dire telle que chaque demi-plan ouvert délimité par cette droite contient au plus $n/2$ points rouges et $m/2$ points bleus.*

Références

- Computational Geometry. Algorithms and applications. de Berg, van Kreveld, Overmars and Schwarzkopf. Springer 1997.

- Algorithms in Combinatorial Geometry. H. Edelsbrunner. Springer Verlag Monographs in TCS. 1987.

- Algorithmic Geometry. J-D Boissonat et M. Yvinec. Cambridge University Press, 1998 (version française : Ediscience international, 1995).

Pour les arrangements et les balayages voir également l'article "Topologically sweeping an Arrangement" de Edelsbrunner et Guibas. On y montre en particulier qu'on peut balayer un arrangement de n droites dans le plan en temps $O(n^2)$. Un balayage simple selon les abscisses des intersections auraient une complexité en $O(n^2 \log n)$. L'idée est ici de remplacer la droite verticale de balayage par une front x-monotone de balayage. Ce front est composé d'un segment par droite dans un ordre "vertical". Chaque fois que l'on balaye un point d'intersection dans l'arrangement on intervertie l'ordre de deux droites. Le problème est de savoir quelles sont les paires de droites qui peuvent s'intersecter dans la suite du balayage (i.e. dont les segments du front ont leur point droit en commun). Pour avoir cette information en temps constant on maintient deux structures annexes : Les arbres d'horizon supérieur et inférieur. La mise à jour de ces arbres après chaque balayage utilise un parcours de ligne d'horizon. Voir aussi les nombreuses applications dans le même article.

Chapitre 6

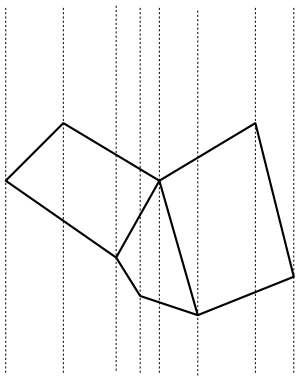
Localisation

Le problème de la localisation consiste, étant donnée une collection fixée de n objets dans $\mathbb{R}^d$, à trouver (les) l'objet(s) de la collection contenant un point de requête quelconque. C'est d'une certaine manière le problème dual de la recherche multidimensionnelle du chapitre 7. Pour $d = 1$, cela revient à trouver les intervalles (ce sont les objets de la collection) contenant le point de requête. Les structures *d'arbre d'intervalles* et *d'arbre de segments* permettent de répondre efficacement à ce type de requête mono-dimensionnelle en temps $O(\log n + k)$ où k est le nombre d'intervalles contenant le point de requête. On pourra consulter [dBCvKO08, chap. 10] pour des détails sur ces structures. Dans ce chapitre on se restreindra au cas où les objets forment une subdivision d'une partie du plan, c'est-à-dire que les régions s'identifient aux faces de la subdivision. Le cas planaire est en particulier utile pour les applications de système d'information géographique.

Les deux sections suivantes sont essentiellement des traductions du chapitre 6 de de Berg et al. [dBCvKO08]

6.1 Localisation par découpe en tranches verticales

On suppose ici l'ensemble des régions donné sous forme d'une carte planaire combinatoire (cf. section 5.2) de taille n, i.e. possédant n segments. Notons que si la carte est connexe, on peut par un simple parcours en temps linéaire marquer chaque demi-arête de la carte avec l'indice de la face située à sa gauche. La localisation consiste alors à reporter l'indice de la face de la carte combinatoire contenant le point de requête. Pour cela on commence par tracer par chaque sommet de la carte une droite verticale comme sur la figure ci-dessous. Ces droites verticales découpent le plan en bandes, ou tranches, verticales. À l'intérieur d'une bande les régions traversées sont ordonnées verticalement. Ainsi pour localiser un point p de requête il suffit d'effectuer deux recherches mono-dimensionnelles : une recherche *horizontale* pour déterminer la bande contenant p, suivie d'une recherche *verticale* permettant d'identifier la région, ou face, de la carte contenant p. Chaque recherche s'effectuant sur un ensemble de taille $O(n)$, la localisation peut s'effectuer en temps $O(\log n)$ à l'aide de structure de dictionnaire classique ou randomisée (cf. chapitre 4).



Plus précisément, pour construire la structure de recherche, on commence par ranger les sommets de la carte dans un dictionnaire du type arbre équilibré en utilisant l'ordre lexicographique sur leurs coordonnées pour les comparaisons. On effectue ensuite un balayage des sommets de la carte de gauche à droite. Dit autrement, on parcourt les sommets dans l'ordre lexicographique. Pour chaque sommet q balayé on construit une structure de recherche $V[q]$ correspondant à la bande verticale située à droite de q. Cette structure de recherche est elle même un dictionnaire sur les arêtes qui traversent $V[q]$ et ordonnées verticalement. Clairement $V[q]$ peut être obtenue à partir du dictionnaire de la bande précédente en ajoutant les arêtes incidentes à q et tournées vers la droite et en supprimant les arêtes incidentes à q et tournées vers la gauche.

Dans le pseudo-code suivant H désigne le dictionnaire des sommets de la carte pour l'ordre lexicographique et $V[q]$ désigne un dictionnaire sur les arêtes traversant la bande verticale à droite du sommet q.

Construction de la structure de recherche
On suppose les sommets $p_0, ..., p_m$ de la carte ordonnés de gauche à droite.

Initialiser un dictionnaire V_{aux} à vide
Pour $i := 0$ à m **faire**
  Insérer p_i dans le dictionnaire H
  $V[p_i] := V_{aux}$
  Pour chaque demi-arête a d'origine p_i **faire**
    Si a est orientée vers la gauche **Alors**
      Supprimer la demi-arête opposée de a dans $V[p_i]$
    Sinon Insérer a dans $V[p_i]$
  $V_{aux} := V[p_i]$

Localisation d'un point p
Rechercher dans H le sommet q de la carte juste à gauche de p
Si $p == q$ **Alors** renvoyer cette information
Sinon Si $p == NILL$ (p est à l'extrême gauche de la carte) **Alors**
  retourner la face la plus à gauche de la carte
Sinon rechercher la demi-arête a (orientée vers la droite) sous p dans $V[q]$
  Si $a == NILL$ **Alors** retourner la face la plus basse de la carte
  Sinon Si p est sur a **Alors** retourner cette information
  Sinon retourner la région à gauche de a

Le temps de construction de la structure de recherche, hormis la recopie des dictionnaires verticaux est $O(n \log n)$: chaque demi-arête est insérée et supprimée une fois dans l'ensemble des bandes au cours du balayage. Cela dit on recopie $O(n)$ fois des structures de tailles $O(n)$ ce qui fournit finalement une construction en temps et espace $O(n^2)$. On peut cependant améliorer ces résultats en utilisant des dictionnaires (arbres) persistants.

Dans un dictionnaire persistant on ordonne dans le temps les opérations d'insertion et de suppression et on peut faire une requête du type : rechercher un élément dans l'état du dictionnaire au moment de la $k$ième opération. On peut obtenir sur n éléments et m opérations un temps d'insertion/suppression en $O(\log n)$, un temps de recherche en $O(\log m)$ et une taille amortie en $O(n)$. Appliqué au présent problème cela donne

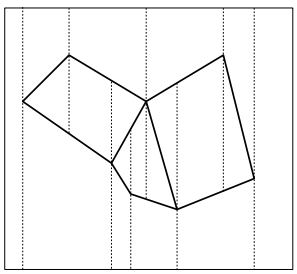
Théorème 6.1 *Soit S un ensemble de n segments formant une carte. L'algorithme exposé calcule en temps $O(n \log n)$ une structure de recherche associée à S de taille $O(n)$. De plus pour tout point p du plan le temps de localisation dans cette structure est $O(\log n)$.*

Référence :

- Éléments d'algorithmique. D. Beauquier, J. Berstel et P. Chrétienne. MASSON, 1992.

6.2 Localisation par décomposition trapézoïdale

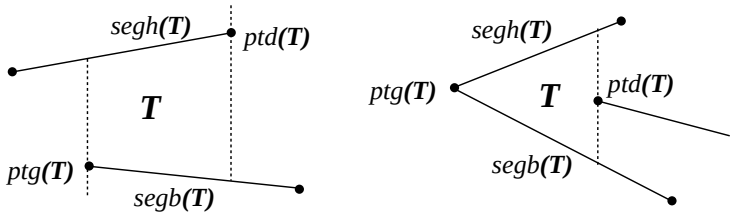
On peut obtenir un temps de recherche équivalent à l'approche par découpe verticale et une structure de recherche de taille linéaire sans utiliser de structure de données persistante. Pour cela on trace comme précédemment une verticale par chaque sommet de la subdivision mais on stoppe cette verticale dès qu'on rencontre un segment de la carte. Afin de simplifier l'exposé, on commence par entourer la carte d'une grande boîte rectangulaire et on ne s'intéresse qu'à l'intérieur de cette boîte. En particulier chaque verticale est bornée (cf. figure ci-dessous). Chaque face de la subdivision ainsi obtenue est convexe : les



angles en chaque sommet d'une telle face (sommet d'arête ou intersection d'une verticale et d'une arête) sont plus petits que π. Chaque face a donc au plus deux côtés verticaux et donc au plus deux non-verticaux et donc exactement deux non-verticaux puisque bornée. Dit autrement chaque face de la subdivision est géométriquement un *trapèze* possédant quatre côtés dont deux verticaux ou trois côtés dont un vertical. La subdivision ainsi obtenue porte ainsi le nom de *carte des trapèzes*.

Un trapèze est entièrement défini par les deux segments qui le bordent en haut et en bas et par les deux sommets qui ont permis de construire ses parois verticales gauche et

droite. On note *segh*, *segb*, *ptg* et *ptd* respectivement ces quatre entités. La figure suivante montre deux exemples de trapèzes avec leurs entités *segh*, *segb*, *ptg* et *ptd*.



On distingue cinq configurations pour le côté vertical gauche (droit) d'une face :

1. il est réduit à un point qui est l'extrémité gauche commune aux segments supérieur et inférieur.
2. c'est l'extension verticale du sommet gauche du segment inférieur qui aboutit sur le segment supérieur.
3. même chose en échangeant supérieur et inférieur.
4. c'est l'extension verticale de l'extrémité droite d'un autre segment qui le coupe donc en deux.
5. c'est un côté de la boîte englobante.

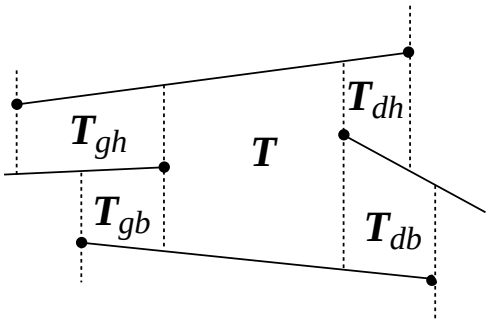
Lemme 6.2 *La carte des trapèzes a un nombre linéaire de faces, arêtes et sommets en fonction du nombre n de segments de la carte planaire d'origine.*

Preuve : Chaque sommet d'origine (au plus $2n$) donne lieu à au plus trois sommets : lui-même + 2 sommets pour les extensions inférieures et supérieures des verticales. Avec les 4 sommets de la boîte englobante cela donne au plus $6n+4$ sommets en tout. De plus chaque sommet d'origine donne lieu à 4 nouveaux segments : ses deux extensions verticales + 2 segments obtenus par subdivision des segments rencontrés par ses extensions verticales. La carte des trapèzes possèdent donc au plus $4.2n+n+4 = 9n+4$ segments en tenant compte de la boîte englobante. La relation d'Euler permet finalement de borner trivialement le nombre de faces par le nombre d'arêtes. □

Exercice 6.3 *Montrer qu'on peut borner le nombre de faces par $3n+1$ en associant judicieusement chaque face à l'un des segments intersectant le bord de cette face.*

On associe à la carte des trapèzes une structure $\mathcal{T}$ la représentant, mais plus simple que la représentation classique en demi-arêtes, et une structure $\mathcal{R}$ de recherche pour localiser un point. Nous supposerons par la suite les sommets de la carte en position générale de sorte que chaque droite verticale passe par au plus un sommet. De cette manière chaque trapèze possède au plus quatre voisins adjacents à ses parois verticales. On note T_{gh}, T_{gb}, T_{dh} et T_{db} ces voisins comme sur la figure ci-dessous.

• La structure $\mathcal{T}$ se présente sous forme d'une liste d'enregistrements correspondant à chacun des trapèzes. Chaque enregistrement contient les pointeurs sur les entités *segh*, *segb*, *ptg*, *ptd*, T_{gh}, T_{gb}, T_{dh} et T_{db} précédemment définies.



• La structure $\mathcal{R}$ est un graphe orienté sans cycle possédant une unique racine et exactement une feuille par trapèze de $\mathcal{T}$. Chaque trapèze de $\mathcal{T}$ pointe sur sa feuille correspondante dans $\mathcal{R}$ et réciproquement[1]. Les noeuds internes de $\mathcal{R}$ ont tous deux enfants et pointent soit sur un sommet soit sur un segment de la subdivision originale. Chaque noeud interne ν est la racine dans $\mathcal{R}$ d'un sous-graphe orienté sans cycle qui est une structure de recherche pour l'intersection d'une certaine zone Z_ν avec $\mathcal{T}$. Cette zone est définie récursivement comme suit. La zone de la racine de $\mathcal{R}$ est l'intérieur de la boîte englobante. Si ν pointe sur un sommet alors la zone de sa fille gauche (droite) est la partie de Z_ν à gauche (droite) de ce sommet. Si ν pointe sur un segment alors la zone de sa fille gauche (droite) est la partie de Z_ν au dessus (au dessous) de ce segment.

Pour effectuer une recherche d'un point p dans la structure de recherche $\mathcal{R}$, on part de la racine et on descend progressivement jusqu'à une feuille qui pointe sur le trapèze contenant p. À chaque noeud pointant sur un sommet on s'oriente vers la fille gauche ou droite selon que p est à gauche ou à droite de ce sommet. De même, à chaque noeud pointant sur un segment on s'oriente vers la fille gauche ou droite selon que p est au dessus ou au dessous de ce segment. La feuille atteinte correspond au trapèze contenant p. Ce trapèze peut lui-même être indéxé par la face le contenant dans la carte d'origine (via la face à gauche d'une demi-arête de son segment inférieur) fournissant ainsi le résultat cherché.

$\mathcal{T}$ et $\mathcal{R}$ sont construit de manière incrémentale randomisée en insérant successivement chaque segment s_i de la subdivision d'origine, pris dans un ordre aléatoire, dans les structures $\mathcal{T}_{i-1}$ et $\mathcal{R}_{i-1}$ associées aux $i-1$ premiers segments insérés.

L'algorithme de construction est le suivant :

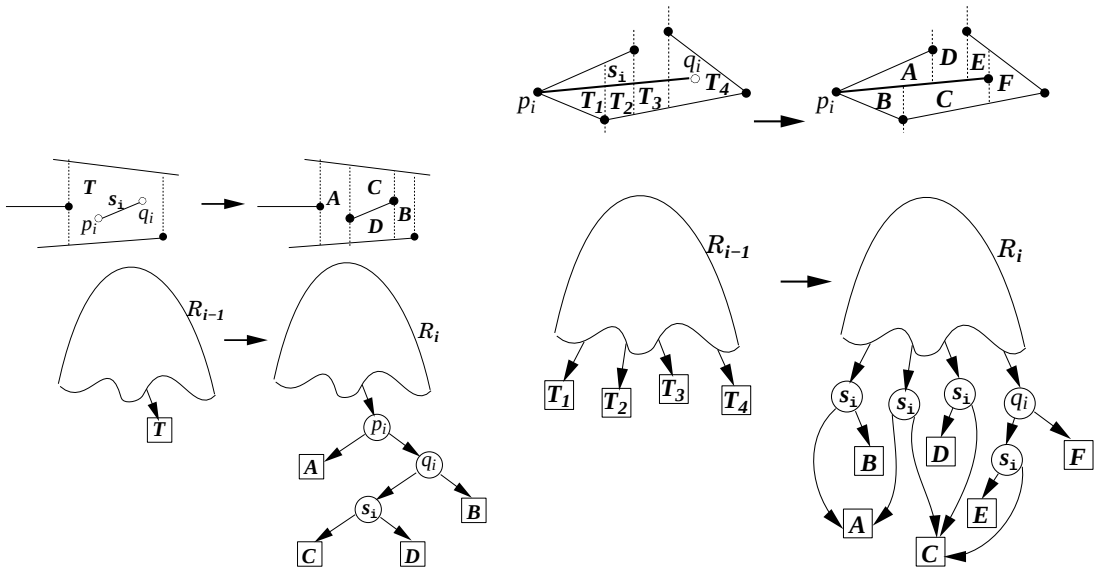
Données : Une carte planaire formée de n segments $s_1, \ldots, s_n$ énumérés dans un ordre aléatoire.
Sorties : Les structures $\mathcal{T}$ et $\mathcal{R}$ associées.

1. Déterminer une boite englobante contenant les n segments (strictement) et initialiser $\mathcal{T}_0$ et $\mathcal{R}_0$
2. **Pour** $i := 1$ à n **faire**
  3. Localiser, grâce à $\mathcal{R}_{i-1}$, le trapèze T_1 de $\mathcal{T}_{i-1}$ contenant l'extrémité gauche de s_i.
  4. En déduire pas simple parcours dans $\mathcal{T}_{i-1}$ les trapèzes $T_1, T_2, \ldots, T_k$ traversés par s_i.
  5. Remplacer chaque trapèze $T_1, T_2, \ldots, T_k$ de $\mathcal{T}_{i-1}$ par sa subdivision, induite par s_i,

[1] D'une autre façon on peut voir $\mathcal{T}$ comme une structure définie sur les feuilles de $\mathcal{R}$

et chacune composée de 2,3 ou 4 trapèzes.
6. Faire de même dans $\mathcal{R}_{i-1}$ en remplaçant cette fois chaque trapèze feuille par un sous-arbre de recherche élémentaire sur sa subdivision.
7. Fusionner les trapèzes des subdivisions pour rétablir la décomposition trapézoïdale induite par $s_1, \ldots, s_i$.
finpour

Pour l'étape 6, il y a essentiellement 3 types de sous-arbres élémentaires possibles selon qu'un trapèze est découpé en 2,3 ou 4 morceaux. Les noeuds internes des ces sous-arbres correspondent au segment s_i ou à ses extrémités. La figure suivante résume les étapes 6 et 7 sur deux exemples.



La structure de recherche $\mathcal{R}$ dépend évidemment de l'ordre d'insertion des n segments de la carte d'origine. On considère $\mathcal{R}$ comme une variable aléatoire sur l'espace des permutations (sur l'insertion) des segments muni de la loi uniforme.

Lemme 6.4 *Soit p un point fixé du plan. Le temps moyen de localisation de p dans $\mathcal{R}$ est un $O(\log n)$.*

Preuve : Le temps de localisation de p est clairement proportionnel à la longueur de son chemin de recherche $l(p)$ dans $\mathcal{R}$.

Remarque : à chaque insertion d'un segment dans la construction incrémentale la hauteur de $\mathcal{R}$ augmente d'au plus 3. Un chemin de recherche a donc une longueur bornée par $3n$.

On note X_i la variable aléatoire dénombrant les noeuds (internes) de $l(p)$ apparus dans $\mathcal{R}$ lors de l'insertion de s_i, i.e. les noeuds de $l(p)$ présents dans $\mathcal{R}_i$ mais pas dans $\mathcal{R}_{i-1}$. En notant Y_i la variable aléatoire valant 1 si ce chemin est rallongé à l'étape i et 0 sinon, on a suivant la remarque précédente

$$|l(p)| = \sum_i X_i \leq 3 \sum_i Y_i.$$

D'où en prenant les espérances

$$E(|l(p)|) \leq 3 \sum_i E(Y_i) = 3 \sum_i P(Y_i = 1).$$

Or $l(p)$ est rallongé à l'étape i si et seulement si le segment s_i intersecte le trapèze contenant p dans la carte $\mathcal{T}_{i-1}$ c.-à-d. si et seulement si le trapèze Δ_i contenant p dans la carte $\mathcal{T}_i$ n'existe pas dans la carte $\mathcal{T}_{i-1}$. Mais $\Delta_i \notin \mathcal{T}_{i-1}$ si et seulement si l'une des 4 entités qui le définissent, ($segh(\Delta_i)$, $segb(\Delta_i)$, $ptg(\Delta_i)$ ou $ptd(\Delta_i)$) n'est pas présente dans $\mathcal{T}_{i-1}$. Selon le principe de *l'analyse arrière* fixons le sous-ensemble S_i des i premiers segments insérés. Notons que $\mathcal{T}_i$ – et donc Δ_i – ne dépend que de S_i et non de l'ordre d'insertion des i premiers segments. On a alors

$$\begin{aligned} P(Y_i = 1 \mid S_i) \quad \leq \quad & P(segh(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) + P(segb(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) \\ & + P(ptg(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) + P(ptd(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) \end{aligned}$$

Clairement $P(segh(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) = P(segh(\Delta_i) = s_i \mid S_i) = 1/i$ et de même $P(segb(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) = 1/i$. Si plusieurs segments de S_i ont $ptg(\Delta_i)$ pour extrémité alors $P(ptg(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i)$ est nul. Sinon, en notant s l'unique segment de S_i d'extrémité $ptg(\Delta_i)$, on a $P(ptg(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) = P(s = s_i \mid S_i) = 1/i$. D'où dans tous les cas $P(ptg(\Delta_i) \notin \mathcal{T}_{i-1} \mid S_i) \leq 1/i$. En résumé,

$$P(Y_i = 1 \mid S_i) \leq 4/i$$

D'où inconditionnellement $P(Y_i = 1) \leq 4/i$. On en déduit $E(|l(p)|) \leq 12 H_n = O(\log n)$. □

Lemme 6.5 *La taille moyenne de la structure $\mathcal{R}$ de recherche est linéaire.*

Preuve : On vérifie que si k_i est le nombre de nouveaux trapèzes créés à l'étape i, alors le nombre de noeuds internes ajoutés à $\mathcal{R}_{i-1}$ est $k_i - 1$: soient $T_1, \ldots, T_k$ les trapèzes intersectés par s_i. Si T_j est subdivisé en $n(T_j)$ morceaux par l'introduction de s_i alors son sous-arbre élémentaire a $n(T_j) - 1$ noeuds internes. Soit un total de $N_i = \sum_j n(T_j) - k$ noeuds internes créés par l'introduction de s_i. Mais le nombre de nouveaux trapèzes est $\sum_j n(T_j) - (k-1) = N_i + 1$ car il faut considérer autant de fusions que de trapèzes intersectés moins 1.

Puisque la carte des trapèzes a une taille linéaire (lemme 6.2) $\mathcal{R}$ possède $O(n)$ feuilles plus $\sum_i (k_i - 1)$ noeuds internes. D'où $E(|\mathcal{R}|) = O(n) + \sum_i E(k_i)$. On utilise à nouveau une analyse arrière : pour S_i fixé, un trapèze de $\mathcal{T}_i$ a une probabilité au plus $4/i$ d'avoir été créé par l'insertion de s_i. Le nombre moyen $E(k_i)$ de trapèzes créés à l'étape i, qui est la somme de cette probabilité pour tous les trapèzes de $\mathcal{T}_i$, est donc majoré par $O(i)4/i = O(1)$ puisqu'il y a $O(i)$ trapèzes sans $\mathcal{T}_i$. □

En ce qui concerne le temps moyen de construction on remarque que le coût d'insertion de s_i est le temps de localisation de son extrémité gauche plus $O(k_i)$ pour parcourir les trapèzes traversés et mettre à jour $\mathcal{R}_{i-1}$ et $\mathcal{T}_{i-1}$. Le temps moyen de localisation est $O(\log i)$ par le lemme 6.4 et $E(k_i) = O(1)$ d'après ce qui précède. On en déduit un coût moyen total de $O(n \log n)$.

En résumé,

Théorème 6.6 *Soit S un ensemble de n segments formant une carte planaire. L'algorithme exposé calcule en temps moyen $O(n \log n)$ une structure de recherche pour la carte des trapèzes associée à S de taille moyenne $O(n)$. De plus pour tout point p du plan le temps moyen de localisation dans cette structure est $O(\log n)$.*

Notons que les moyennes sont ici prises relativement aux permutations sur les segments et que p est fixé une fois pour toute. En particulier, ce résultat ne dit rien sur le temps maximal moyen relativement à l'ensemble des points requêtes. Autrement dit il se pourrait que pour la plupart des permutations des segments il existe un point, dépendant de la permutation considérée, ayant un "mauvais" temps de localisation. En fait le temps maximal moyen est lui-même majoré par $O(\log n)$. Pour le voir on commence par remarquer que deux points appartenant à la même cellule de l'arrangement des droites supports des n segments et des $2n$ verticales aux sommets des segments partagent un même chemin de recherche dans $\mathcal{R}$ quelque-soit la permutation considérée. Pour chaque permutation, il y a donc $O(n^2)$ chemins de recherche distincts possibles[2] qui ont bien sûr chacun une longueur moyenne en $O(\log n)$. Si on montre que pour chaque chemin cette moyenne n'est dépassée que pour un sous-ensemble suffisamment petit de permutations alors ce sera encore le cas pour le maximum des longueurs de ces $O(n^2)$ chemins. C'est l'objet de ce qui suit.

Pour faire une analyse plus fine de la queue de distribution on utilise la technique de Chernoff. Le problème est ici que les variables aléatoires Y_i introduites plus haut ne sont pas indépendantes (exercice : le vérifier pour S formé des cotés d'un triangle et un point interne à ce triangle). On utilise une définition modifiée qui les rend indépendantes. Pour cela on considère le diagramme de Hasse du treillis (booléen) des parties de S pour l'inclusion. Ce treillis a $\binom{n}{i}$ noeuds de *niveau* i ayant chacun i arcs entrants et $n-i$ arcs sortants. Les permutations de S sont en bijection avec les chaînes maximales de ce treillis (joignant vide à plein) et chaque arc du treillis correspond à un segment de S (le segment ajouté). On fixe un point p et on marque un arc si la *suppression* du segment correspondant modifie le trapèze contenant p dans la carte associée à la tête de cet arc. On a vu que chaque noeud a au plus 4 arcs entrants marqués. L'astuce est de compléter systématiquement à 4 le nombre d'arcs entrants marqués (pour les noeuds de niveaux 1, 2, 3 on marque tous les arcs entrants). Le point p étant fixé, on note X_i la variable aléatoire définit sur les chaînes maximales du diagramme (donc les permutations) valant 1 si l'arc entre les niveaux $i-1$ et i est marqué et 0 sinon. Ainsi la longueur du chemin de recherche de p est majorée par $3\sum_i X_i$.

Lemme 6.7 *Les X_i sont mutuellement indépendantes.*

Preuve : Commençons par remarquer que si A est une condition portant sur des niveaux supérieurs ou égaux à i alors $P(X_i = \epsilon \mid A)$ ne dépend pas de A et vaut donc $P(X_i = \epsilon)$. Par exemple, si a est une arête du treillis entre les niveaux i et $i+1$, la probabilité que $X_i = 1$ restreinte aux permutations contenant a vaut $4/i$ si $i \geq 4$.

On raisonne ensuite par récurrence sur le nombre k de v.a. prises en compte. On suppose que les k premières variables aléatoires $X_1, \ldots, X_k$ sont mutuellement indépendantes et

[2] Voir le théorème 5.3 sur les arrangements de droites.

plus précisément que pour toute condition A_k portant sur des niveaux $\geq k$, les v.a. $X_1|A_k, \ldots, X_k|A_k$ le sont avec des probabilités uniformes par rapport à A_k. Dit autrement, pour tout $\epsilon_1, \ldots \epsilon_k \in \{0,1\}^k$, on a

$$P(X_1 = \epsilon_1 \wedge \ldots \wedge X_k = \epsilon_k \mid A_k) = \Pi_{j=1}^{k} P(X_j = \epsilon_j) \tag{6.1}$$

Notons B l'événement $(X_1 = \epsilon_1 \wedge \ldots \wedge X_k = \epsilon_k)$. On a alors (cf. exercice 1.19) pour toute condition A_{k+1} portant sur des niveaux $\geq k+1$ et pour tout $\epsilon_{k+1} \in \{0,1\}$:

$$P(B \wedge X_{k+1} = \epsilon_{k+1} \mid A_{k+1}) = P(B \mid X_{k+1} = \epsilon_{k+1} \wedge A_{k+1}) P(X_{k+1} = \epsilon_{k+1} \mid A_{k+1})$$

Ce qui vaut encore $\Pi_{j=1}^{k+1} P(X_j = \epsilon_j)$ d'après l'équation (6.1) et la remarque initiale. □

On peut maintenant appliquer la technique de Chernoff (cf. section 1.7.4) à la variable $X_p = \sum_i X_i$. Pour cela on regarde la probabilité que X_p dépasse $\lambda \ln(n+1)$. En utilisant l'inégalité de Markov on a

$$\forall t > 0,\ E(X_p \geq \lambda \ln(n+1)) = E((e^{tX_p}) > (n+1)^{t\lambda}) \leq E(e^{tX_p})/(n+1)^{t\lambda}$$

Or de part l'indépendance des X_i on a

$$E(e^{tX_p}) = \prod_i E(e^{tX_i})$$

Avec $t = \ln(5/4)$ on calcule pour $i \geq 4$, $E(e^{tX_i}) = (5/4)4/i + 1(1 - 4/i) = (i+1)/i$ et $E(e^{tX_i}) = 5/4$ pour $i = 1, 2, 3$. D'où $E(e^{tX_p}) \leq n+1$. Finalement on trouve

$$E(X_p \geq \lambda \ln(n+1)) \leq 1/(n+1)^{\lambda \ln(5/4) - 1}$$

On a vu plus haut que X_p ne dépend que de la cellule d'un certain arrangement de taille $O(n^2)$. La probabilité que l'un des X_p définis pour chacune des cellules de l'arrangement dépasse $\lambda \ln(n+1)$ est donc majorée par $O(1/(n+1)^{\ln(5/4)\lambda - 3})$. Comme X_p est uniformément borné par $O(n)$ on en déduit que

$$\begin{aligned} E(\max_p X_p) &= \sum_k k P(\max_p X_p = k) \\ &\leq \lambda \ln(n+1) P(\max_p X_p < \lambda \ln(n+1)) + O(n) P(\max_p X_p \geq \lambda \ln(n+1)) \\ &\leq \lambda \ln(n+1) + O(1/(n+1)^{\lambda \ln(5/4) - 4}) \end{aligned}$$

Finalement, en choisissant λ tel que $\lambda \ln(5/4) \geq 4$, on obtient

Lemme 6.8 *L'espérance du temps* maximal *de recherche est* $O(\log n)$.

On montrerait de même que la probabilité que la taille (resp. le temps de construction) ne dépasse pas λn (resp. $\lambda n \log n$) est aussi proche de 1 que l'on veut si on choisit λ assez grand. Du coup la probabilité P que le temps maximal de recherche, la taille, et le temps de construction soient simultanément comme désirés est non nulle (et en fait P est aussi proche de 1 que l'on veut si on fait croître λ). On en déduit

Théorème 6.9 *Soit S l'ensemble des n segments d'une carte planaire. On peut construire en temps moyen $O(n \log n)$ une structure de recherche pour la carte des trapèzes associée à S de taille $O(n)$ et de temps de requête $O(\log n)$ dans le cas le pire.*

Pour cela on déroule l'algorithme précédent pour une permutation aléatoire jusqu'à s'apercevoir que la taille ou la profondeur de $\mathcal{R}$ dépasse λn ou $\lambda \log n$. Si c'est le cas on recommence avec une autre permutation aléatoire jusqu'à obtenir les taille et profondeur désirées. La moyenne du nombre d'essais à effectuer est $1/P$ et le temps moyen de construction est donc $O(n \log n)/P = O(n \log n)$.

Référence :

- Computational Geometry. Algorithms and applications. de Berg, van Kreveld, Overmars and Schwarzkopf. Springer 1997. pf. Springer 1997.

6.3 Localisation dans une triangulation

Une autre approche, proposée par Kirkpatrick [Kir83], consiste à partir d'une triangulation. Cette dernière peut être obtenue pour une carte planaire de taille n en temps $O(n \log n)$ suivant les algorithmes présentés au chapitre 3. La méthode de localisation de Kirkpatrick repose sur une représentation hiérarchique de la triangulation obtenue par simplifications progressives. Pour passer d'une triangulation à une triangulation simplifiée on supprime un ensemble de sommets indépendants (i.e. tel que deux de ces sommets ne soient pas adjacents) et de degré borné puis on retriangule si nécessaire les "trous" laisser par les sommets supprimés. Pour que cet algorithme soit efficace on cherche à construire des ensembles de sommets indépendants, et de degré borné, les plus grands possibles. La borne sur le degré assure que chaque triangle à un niveau de la hiérarchie intersecte un nombre borné de triangles du niveau suivant.

Dans ce qui suit le terme de *triangulation* désigne une triangulation rectiligne du plan dont la face externe est elle-même un triangle (c'est donc combinatoirement une triangulation de la sphère). On peut toujours se ramener à une telle triangulation à partir d'une triangulation quelconque d'un domaine convexe en ajoutant trois sommets "à l'infini" et en joignant convenablement les sommets du bord du domaine à ces trois sommets. Les sommets d'une triangulation distincts des 3 sommets de la face externe sont dits *internes*.

Lemme 6.10 *Soit $\mathcal{T}$ une triangulation ayant n sommets internes et un entier $d \geq 6$. On peut sélectionner en temps $O(n)$ un nombre αn de sommets internes, indépendants et de degrés au plus d avec $\alpha = \frac{d-5}{(d+1)(d-2)}$.*

Preuve : On considère l'algorithme glouton suivant : parcourir les sommets internes de $\mathcal{T}$ et sélectionner le sommet courant du parcours si son degré est au plus d et si aucun de ses voisins n'est sélectionné. L'algorithme glouton sélectionne clairement un ensemble de sommets internes, indépendants et de degrés au plus d en temps linéaire. Reste à minorer sa taille n_s.

Soit x et y les nombres de sommets internes de $\mathcal{T}$ respectivement de degré au plus d et au moins $d+1$. En particulier, $x + y = n$. On note a le nombre d'arêtes de $\mathcal{T}$. Par la

relation d'Euler et la relation d'incidence arête/face on obtient aisément $a = 3n + 3$. De la relation d'incidence arête/sommet on déduit $2a = \sum_s \text{degré}(s)$, la somme portant sur tous les sommets de $\mathcal{T}$. En coupant cette somme en trois selon les sommets internes de degrés inférieurs ou supérieurs à d et les 3 sommets de la face externe, et en remarquant que tout sommet a degré 3 au moins, on obtient

$$2a \geq 3x + (d+1)y + 9$$

Ce qui, avec les relations précédentes implique

$$x \geq \frac{(d-5)n+3}{d-2} \geq \frac{(d-5)n}{d-2}$$

Mais l'algorithme glouton sélectionne un ensemble maximal de sommets. Dit autrement, l'ensemble des sommets sélectionnés et leurs voisins couvrent les sommets internes de degrés au plus d. On en déduit $(d+1)n_s \geq x$, soit encore

$$n_s \geq \frac{d-5}{(d+1)(d-2)}n$$

□

On construit une suite de triangulations $\mathcal{T} = \mathcal{T}_1, \ldots, \mathcal{T}_m$ ayant respectivement $n = n_1, \ldots, n_m$ sommets internes avec $n_m = 0$. On obtient $\mathcal{T}_{i+1}$ à partir de $\mathcal{T}_i$ en 2 étapes :

1. on sélectionne dans $\mathcal{T}_i$ un ensemble maximal de sommets internes, indépendants et de degré au moins d par l'algorithme glouton ci-dessus,
2. on supprime ces sommets de $\mathcal{T}_i$ et on retriangule l'étoile de ces sommets (de taille d) en temps constant par étoile.

Le lemme précédent montre que $n_{i+1} \leq (1-\alpha)n_i$, d'où $m = O(\log n)$. On ajoute de plus à chaque triangle de $\mathcal{T}_{i+1}$ des pointeurs vers les au plus $d-2$ triangles de $\mathcal{T}_i$ qu'il intersecte. Clairement cette structure est de taille $O(\sum_i n_i) = O(n)$ et peut être construite dans le même temps.

On détermine le triangle de $\mathcal{T}$ contenant un point p de requête par les étapes suivantes

1. déterminer si p est dans l'unique triangle de $\mathcal{T}_m$,
2. pour i allant de $m-1$ à 1, connaissant le triangle de $\mathcal{T}_{i+1}$ contenant p, déduire celui de $\mathcal{T}_i$ contenant p à l'aide des pointeurs décrits ci-dessus.

Chacune des $m = O(\log n)$ étapes ci-dessus prend un temps constant puisque le nombre de pointeurs à tester entre deux triangulations est majoré par $d-2$.

Finalement,

Théorème 6.11 *Étant donné une triangulation $\mathcal{T}$ du plan, on peut construire en temps linéaire en la taille de $\mathcal{T}$ une structure de taille linéaire, permettant de localiser un point quelconque en temps logarithmique.*

Chapitre 7

Recherche multidimensionnelle

Soit d un entier positif. On se donne un ensemble S de n points de $\mathbb{R}^d$ et un ensemble $\mathcal{B}$ de parties de $\mathbb{R}^d$ appelées boîtes. Le problème de la recherche multidimensionnelle est de répondre à une requête du type :

Compter ou reporter les points de S contenus dans une boîte donnée de $\mathcal{B}$.

Pour répondre le plus rapidement possible à ce type de requête on construit généralement une structure de recherche qui aidera à répondre à toutes les requêtes possibles. Cette construction est donc considérée comme un pré-calcul dont le temps d'exécution peut raisonnablement être important par rapport au temps de réponse de chaque requête individuelle. La taille de cette structure doit par contre rester relativement faible dans la mesure où elle persiste pour toutes les requêtes. Dans la pratique un algorithme possédant un faible temps de requête nécessitera une structure plus volumineuse qu'un algorithme moins performant. Un exemple extrême consiste en une structure de taille linéaire se limitant à la liste des points de S et un algorithme de recherche au mieux linéaire obtenu en testant individuellement si chaque point de S est contenu dans la boîte de requête. Le problème général de la recherche multidimensionnelle suppose ainsi de faire des compromis entre l'espace requis pour la structure de donnée et la rapidité de réponse à une requête.

Références :

- [dBCvKO08, Chap. 5 and 16], [Mat94], [AE99], [GO04, Chap. 36].

7.1 Recherche orthogonale

La recherche orthogonale désigne le cas particulier de la recherche multidimensionnelle où l'ensemble des boîtes $\mathcal{B}$ est l'ensemble (infini) des pavés de $\mathbb{R}^d$.

Une approche directe consiste à pré-calculer toutes les réponses possibles ($\subset \mathcal{P}(S)$) et trouver un critère sur les boîtes pour savoir quand elles donnent la même réponse. Par ce moyen on obtient des temps de requêtes très performants mais des temps de pré-calculs très importants – ce qui n'est pas forcément rédhibitoire – et surtout des stockages très importants - ce qui est plus ennuyeux car ils persistent.

Exemple dans le plan : on trace une verticale et une horizontale par chaque point de S de manière à obtenir une grille. Deux boîtes donnent la même réponse si et seulement si leurs coins supérieurs gauches et inférieurs droits sont dans les mêmes cases de la grille. On a donc $\binom{n+1}{2}^2$ réponses non-vides possibles (une boîte est déterminée par deux verticales et 2 horizontales). Comme chaque réponse peut contenir $O(n)$ points, on obtient naïvement une structure de stockage de taille $O(n^5)$ pour le problème du report des points.

On caractérise un algorithme de recherche par la paire (taille stockage, temps de requête).

7.1.1 Recherche unidimensionnelle

En dimension 1, le problème de la recherche orthogonale revient à compter ou reporter tous les points de S contenus dans un intervalle $[x, x']$ de requête.

Pour ce faire, on utilise un arbre binaire de recherche équilibré (la hauteur des sous-arbres gauche et droit de tout noeud diffère de un au plus) avec les points rangés aux feuilles, les noeuds internes contenant des valeurs de séparation entre les clés des sous-arbres gauche et droit. On suppose également que les feuilles sont chaînées dans l'ordre croissant des coordonnées des points. Pour le problème du report, on peut chercher dans l'arbre la feuille contenant x puis marcher jusqu'à x' en utilisant le chaînage. Ceci fournit une réponse en temps $(\log n + k)$, où k est le nombre de points à reporter. Cette méthode se généralise difficilement en dimension supérieure. Pour y remédier on commence par introduire la notion suivante :

Définition 7.1 *L'ensemble des clés des feuilles associées à un sous arbre de racine ν s'appelle l'*ensemble canonique *de ν et est noté $P(\nu)$.*

Plutôt que d'utiliser un chaînage aux feuilles on travaille avec la notion d'ensemble canonique. On commence ainsi par rechercher les feuilles contenant x et x' dans l'arbre de recherche. Les chemins de recherche de x et x' partent de la racine, restent confondus sur une certaine longueur, puis se séparent en deux sous-chemins γ_x et $\gamma_{x'}$ à partir d'un noeud ν_s. Il est facile de voir que l'ensemble cherché est l'union des ensembles canoniques des enfants droits (resp. gauches) des noeuds internes du chemin γ_x (resp. $\gamma_{x'}$). À cette union, on ajoute éventuellement les feuilles de γ_x et γ'_x suivant les comparaisons de x ou x' relativement à ces feuilles. Notons que le calcul de l'ensemble canonique $P(\nu)$ d'un noeud ν peut s'obtenir en temps proportionnel à sa taille $|P(\nu)|$ puisque la taille d'un arbre binaire équilibré est proportionnelle à son nombre de feuilles. En résumé,

Lemme 7.2 (Le problème de la recherche orthogonale en dimension 1) *En utilisant un arbre binaire de recherche on obtient :*
pré-calcul : $O(n \log n)$
espace : $O(n)$
temps requête : $O(\log n + k)$
où k est le nombre de points à reporter.

Exercice 7.3 *Modifier l'algorithme pour le problème du comptage seul et non du report des points eux-mêmes. Quelle est la complexité d'une requête ?*

Voir également : arbres d'intervalles, arbre de recherche de priorité et arbres de segments.

7.1.2 Kd-trees (arbres k-dimensionnels) (Bentley 1975)

Pour résoudre le problème en dimension $d > 1$, on construit récursivement un arbre binaire équilibré tel que pour chaque noeud ν de profondeur k, les ensembles canoniques de ses noeuds enfants constituent une partition de $P(\nu)$ selon la valeur médiane de la (k mod d)-ème coordonnée. Ainsi, l'ensemble canonique de l'enfant gauche (resp. droit) de ν est le sous-ensemble des points de $P(\nu)$ dont la (k mod d)-ème coordonnée est majorée (resp. strictement minorée) par cette valeur médiane.

Remarque : la valeur médiane des points ordonnés selon une coordonnée particulière peut se calculer en temps linéaire. On peut soit utiliser d listes – une par coordonnées – triées au départ et que l'on décime au fur et à mesure que l'on descend dans l'arbre, soit utiliser l'algorithme linéaire classique de calcul de médiane rappelé ci-dessous :

1. Choisir un pivot soit aléatoirement soit par la méthode suivante : couper la liste en $n/5$ groupes de 5 clés et calculer la médiane de chaque groupe, puis calculer récursivement la médiane de ces médianes,
2. scinder la liste en deux selon cette valeur,
3. itérer sur la sous-liste contenant la vraie médiane (i.e. la valeur de rang milieu).

Complexité : $T(n) = O(n)+T(n/5)+T(7n/10) = O(n)$. En effet la plus petite sous-liste contient au moins $3 * 1/2 * n/5$ éléments et du coup la plus grande en contient au plus $7n/10$.

Dans le plan, le temps $C(n)$ de construction d'un $2d$-arbre est donné par

$$C(n) = \begin{cases} O(1) & \text{si } n = 1 \\ O(n) + 2C(\lceil n/2 \rceil)) & \text{sinon} \end{cases} \tag{7.1}$$

On en déduit un temps de construction en $O(n \log n)$.

Pour effectuer la recherche on remarque que chaque noeud ν de l'arbre correspond à une boîte $B(\nu)$ de la forme $[x_1, x_1'] \times [x_2, x_2']$ avec $x_1, x_1', x_2, x_2' \in \overline{\mathbb{R}}$. En effet la racine correspond à tout le plan et les enfants d'un noeud de profondeur k correspondent à la boîte de ce noeud coupée par le plan d'équation $x_{k \bmod 2} = x_{med}$ où x_{med} est la valeur médiane de séparation associée à ce noeud. De plus les boîtes de tous les noeuds de même profondeur forment une partition de l'espace et de ce fait leurs ensembles canoniques constituent une partition de S.

Pour rechercher tous les points dans une boîte B on descend donc à partir de la racine, puis
- si la boîte B intersecte, sans la contenir, la boîte $B(\nu_g)$ (resp. $B(\nu_d)$) de l'enfant gauche (resp. droit) du noeud courant on continue la recherche sur cet enfant,
- sinon si $B(\nu_g)$ (resp. $B(\nu_d)$) est contenu dans B on renvoie l'ensemble canonique $P(\nu_g)$ (resp. $P(\nu_d)$).

Le temps de recherche est égal au nombre de noeuds visités plus le temps pour rendre les $P(\nu)$. Ce dernier est proportionnel au nombre k de points retournés. Pour majorer le

nombre de noeuds internes visités on remarque que ceux-ci ont leur boîte intersectée par le bord de B et donc par les droites supports de B. En majorant le nombre de boîtes $B(\nu)$ intersectées par une droite donnée de direction horizontale ou verticale on obtient donc un majorant du nombre de noeuds internes visités.

Fixons une droite verticale D. On note $I(n)$ le nombre maximal de noeuds internes dont la boîte intersecte D dans un $2d$-arbre quelconque de taille n. Puisque la racine d'un $2d$-arbre est associée à une dichotomie verticale, la droite D ne peut donc intersecter que 2 des 4 boîtes associées aux noeuds de profondeur 2. Comme les noeuds de profondeur 2 sont également associés à des dichotomies verticales et eux-mêmes racines de $2d$-arbres de taille au plus $\lceil n/4 \rceil$, on peut écrire

$$I(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ 2 + 2I(\lceil n/4 \rceil)) & \text{sinon} \end{cases} \tag{7.2}$$

D'où, suivant la section 1.9 ou suivant un calcul directe sur l'arbre de récursion, $I(n) = O(\sqrt{n})$. Un raisonnement analogue permet de borner le nombre de noeuds intersectés par une droite horizontale.

En résumé

Lemme 7.4 *Le problème de la recherche orthogonale dans le plan peut se résoudre en utilisant un $2d$-arbre avec :*
pré-calcul : $O(n \log n)$
espace : $O(n)$
temps requête : $O(\sqrt{n} + k)$ *où k est le nombre de points à reporter.*

Exercice 7.5 *Généraliser à $d > 2$ dimensions. Montrer en particulier que le temps de recherche est en $O(n^{1-1/d} + k)$.*

7.1.3 Arbres de domaines

Les arbres de domaines (*range trees*), introduits par Bentley en 1979, sont des structures de partitionnement de l'espace à plusieurs niveaux. En dimension 2, on commence par construire un arbre binaire de recherche équilibré selon la première coordonnée, puis pour chaque noeud ν on construit un arbre binaire de recherche équilibré sur l'ensemble canonique de ce noeud selon la deuxième coordonnée. En remarquant que les ensembles canoniques $P(\nu)$ correspondant à des noeuds de profondeur donnée forment une partition de l'ensemble des points, et que la hauteur de l'arbre de recherche "primaire" est en $O(\log n)$ en voit que la taille d'un arbre de domaines est en $O(n \log n)$. Pour la construction on commence par trier les points selon la deuxième coordonnées y. Puis on crée une racine en la faisant pointer sur un arbre binaire de recherche selon la deuxième coordonnée. Cet arbre binaire de recherche est construit en temps linéaire à partir de la liste triée de manière "bottom-up". On scinde alors la liste en deux selon x et on construit les deux sous-listes triées en y à partir de la liste triée de départ. Puis on continue récursivement. De cette manière le temps de construction est proportionnel à la taille de la structure finale (chaque arbre "secondaire" associé à un ensemble canonique est construit en temps linéaire).

Pour effectuer une requête avec une boîte $B = [x, x'] \times [y, y']$ on commence par rechercher les valeurs x et x' dans l'arbre primaire, puis pour chaque sous-arbre à droite (gauche) du chemin de recherche γ_x ($\gamma_{x'}$) situé après le noeud de séparation on fait une recherche en y dans la structure secondaire associée à sa racine. Le temps de la recherche est donc majoré par

$$\sum_{\nu} O(\log n + k_\nu) = O(\log^2 n + k)$$

où ν décrit les racines des sous-arbres sus-cités et k_ν le nombre de points reportés dans les ensembles canoniques associés.

Lemme 7.6 *Le problème de la recherche orthogonale dans le plan peut se résoudre en utilisant un arbre de domaines de dimension 2 avec :*
pré-calcul : $O(n \log n)$
espace : $O(n \log n)$
temps requête : $O(\log^2 n + k)$ *où k est le nombre de points à reporter.*

Remarque : Par rapport aux $2d$-arbres on a diminué le temps de requête mais augmenter la taille de la structure.

Exercice 7.7 *Généraliser les arbres de domaines à $d > 2$ dimensions.*

Exercice 7.8 *Jusqu'à maintenant, on a implicitement supposé que les k-ièmes coordonnées des points étaient deux à deux distinctes pour chaque k. Si ce n'est pas le cas on peut considérer la transformation*
$(x, y) \mapsto ((x, y), (y, x))$ *et utiliser l'ordre lexicographique sur les couples (x, y). La boîte de requête devient alors $[(x, -\infty), (x', \infty)] \times [(y, -\infty), (y', \infty)]$. Généraliser ce procédé à la dimension d.*
Remarque : cette astuce revient à perturber les données, c'est à dire à opérer une transformation du type $(x, y) \mapsto (x + \epsilon y, y + \epsilon x)$ où $\epsilon |y_i|_{max} \leq \min_{x_i \neq x_j} |x_i - x_j|$ et de même pour y.

Exercice 7.9 *Si on permet aux points d'avoir une partie de leurs coordonnées identiques on peut s'intéresser à tous les points ayant certaines de leurs coordonnées fixées par une requête. On parle alors de requête d'identification partielle (partial match query).*

1. Montrer qu'avec un $2d$-arbre on peut répondre à une requête d'identification partielle en temps $O(\sqrt{n} + k)$ où k est le nombre de points à reporter.

2. Trouver une structure de données utilisant un espace linéaire et qui répond à une requête d'identification partielle en temps $O(\log n + k)$.

3. Montrer qu'avec un kd-arbre de dimension d (i.e. un dd-arbre) on peut répondre à une requête d'identification partielle sur $s < d$ coordonnées en temps $O(n^{1-s/d} + k)$.

4. Trouver une structure de données pour répondre à une requête d'identification partielle en temps $O(d \log n + k)$ en dimension d, si on permet que la structure soit de taille $O(d2^d n)$.

7.1.4 Fractionnement en cascade

Le fractionnement en cascade (Lueker 1978 et Willard 1978) est une modification des arbres de domaines qui permet de gagner un facteur $\log n$ dans le temps de requête. On s'intéresse à la dimension 2. Remarquons que dans un arbre binaire de recherche selon la première coordonnée on a pour tout noeud ν : $P(\nu) \subset P(\text{parent}(\nu))$. On remplace les arbres binaires de recherche en la deuxième coordonnée sur $P(\nu)$ utilisés par les arbres de domaines par des listes triées (selon la deuxième coordonnée) où chaque élément pointe dans les deux listes des noeuds enfants sur la plus petite "valeur" qui lui est supérieure ou égale.

La taille de la structure est la même que celle d'un arbre de domaines, $O(n \log n)$, et elle peut être construite dans le même temps.

Pour effectuer une requête avec une boîte $B = [x, x'] \times [y, y']$ on procède comme pour un arbre de domaines en déterminant les chemins de recherches γ_x et $\gamma_{x'}$. On recherche ensuite la plus petite valeur supérieure à y dans la liste du noeud de séparation de ces chemins et l'on propage en temps constant cette valeur à l'aide des pointeurs précédemment définis. Dans chaque noeud dont on doit reporter l'ensemble canonique on marche alors à partir de cette valeur jusqu'à la plus grande valeur inférieure ou égale à y'. On en déduit un temps de requête en

$$\sum_{\nu} (O(1) + k_\nu) = O(\log n + k)$$

où ν et k_ν sont définis comme pour les arbres de domaines.

Lemme 7.10 *Le problème de la recherche orthogonale dans le plan peut se résoudre en utilisant le fractionnement en cascade avec :*
pré-calcul : $O(n \log n)$
espace : $O(n \log n)$
temps requête : $O(\log n + k)$ *où k est le nombre de points à reporter.*

Exercice 7.11 *Généraliser la technique du fractionnement en cascade en dimension $d > 2$. Montrer en particulier qu'on obtient un temps de recherche en $O(\log^{d-1} n + k)$.*

Note : Le problème de la recherche othogonale est un des plus étudiés dans le domaine. Étant donné l'interaction entre temps de requête et taille de la structure, B. Chazelle [Cha90] a étudié la taille minimale requise si on impose un temps de requête avec report en temps $O(k + \mathit{polylog}(n))$. Autrement dit, on impose un temps de réponse proportionnel à cette dernière plus un polylog[1]. Chazelle étudie cette question dans le cadre d'une variante de machine à pointeurs et montre que toute structure de recherche en dimension d nécessite un espace $\Omega(n(\log n / \log\log n)^{d-1})$.

[1] $\mathit{polylog}(n)$ désigne une fonction de la forme $P(\log n)$ pour un polynôme P fixé.

7.2 Recherche simpliciale et par demi-espace

S désigne à nouveau un ensemble de n points de $\mathbb{R}^d$ fixé une fois pour toute. Le problème de la recherche simpliciale (resp. par demi-espace) consiste à reporter, ou compter, tous les points de S contenus dans un simplexe (resp. un demi-espace) de requête.

7.2.1 Arbre de partitions

L'idée est de partitionner les points en sous-ensembles de tailles approximativement égales inclus dans des domaines dont la description est simple de sorte que la boîte de requête intersecte une faible proportion de ces domaines. On procède alors récursivement sur les points inclus dans chaque domaine pour définir un arbre de partitions.

On regarde le problème dans le plan.

Définition 7.12 *Une* partition simpliciale *d'un ensemble S de n points du plan est une famille de couples $\{(S_i, \Delta_i)\}_{i\in I}$ où les S_i sont des parties de S et les Δ_i des triangles du plan de sorte que les S_i forment une partition de S et que pour chaque i on a $S_i \subset \Delta_i$. Notons que les triangles Δ_i peuvent se chevaucher et qu'un point de S peut appartenir à plusieurs Δ_i. Le nombre $|I|$ de parties est la* taille *de la partition.*

Soit $r \geq 1$. Une partition simpliciale de paramètre r, *ou r-partition, sur S est une partition simpliciale de S dont chaque sous-ensemble S_i contient entre n/r et $2n/r$ points. Notons que la taille d'une r-partition est comprise entre $r/2$ et r.*

Le nombre de croisements *d'une droite avec une partition simpliciale est le nombre de triangles de la partition intersectés par cette droite. Le* nombre de croisements *de la partition est le maximum de ce nombre sur toutes les droites du plan.*

Le théorème 8.9 de la partition simpliciale affirme que pour toute constant r on peut construire en temps $O(n)$ une r-partition de S de nombre de croisements $O(\sqrt{r})$.

On construit récursivement un *arbre de partitions* de paramètre r sur une ensemble S de n points en associant à la racine un nombre d'enfants en correspondance avec les sous-ensembles d'une partition simpliciale de S de paramètre r fixé. On stocke les descriptions des triangles Δ_i de la partition au niveau des enfants puis on continue récursivement sur chaque enfant avec son sous-ensemble S_i associé tant que ce sous-ensemble compte au moins $2r$ points.

Lemme 7.13 *Soit $r > 2$. Un arbre de partitions de paramètre r sur un ensemble de n points a une taille linéaire (i.e. en $O(n)$) et peut être construit en temps $O(n \log n)$.*

Preuve : Puisque chaque partition simpliciale utilisée est de paramètre r, le degré de chaque noeud interne de l'arbre est au moins $r/2$. Soit n_I et n_E les nombres respectifs de noeuds internes et externes de l'arbre de partitions. En comptant de deux façons différentes le nombre d'arêtes de l'arbre de partitions à partir de l'extrémité respectivement inférieure ou supérieure de chaque arête, on obtient :

$$(r/2)n_I \leq n_I + n_E - 1.$$

Soit

$$n_I \leq \frac{2n_E - 2}{r - 2}.$$

Or chaque feuille de l'arbre est associée à au moins un point et les ensembles associés aux feuilles sont disjoints. On en déduit $n_E \leq n$ et par suite $T(n) = O(n)$, puisque r est constant.

Le temps de construction $C(n)$ vérifie d'après le théorème 8.9 de la partition simpliciale

$$C(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(n) + \sum_\nu C(n_\nu) & \text{sinon} \end{cases} \tag{7.3}$$

où ν parcourt les enfants de la racine. Comme $n_\nu \leq 2n/r$, l'arbre de partitions a une hauteur $O(\log n)$. Par ailleurs, l'ensemble des noeuds de profondeur donnée forme une partition des n points et peut donc être traité en temps $O(n)$. On en déduit $C(n) = O(n \log n)$. □

Pour répondre à une requête du type demi-plan l^+ bordé par une droite l on part de la racine puis on teste si chacun des triangles des (au plus r) enfants est contenu dans l^+, disjoint de l^+, ou intersecte l. Dans les deux premiers cas on sait quoi faire sinon on "récurre" sur les triangles enfants intersectés. La complexité $R(n)$ de la recherche est donnée par

$$R(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(r) + \sum_\nu R(n_\nu) & \text{sinon} \end{cases} \tag{7.4}$$

Soit, compte tenu du paramètre r de la partition simpliciale :

$$R(n) \leq cr + c\sqrt{r}R(2n/r).$$

pour une certaine constante c. On en déduit (par le master theorem) $R(n) = O(n^{\log_{r/2} c\sqrt{r}})$. En choisissant $r = \lceil 2(c\sqrt{2})^{1/\epsilon} \rceil$ on a

$$\log_{r/2} c\sqrt{r} = \frac{\log(c\sqrt{r})}{\log(r/2)} = \frac{1}{2} + \frac{\log(c\sqrt{2})}{\log(r/2)} \leq \frac{1}{2} + \frac{\log(c\sqrt{2})}{\log((c\sqrt{2})^{1/\epsilon})} \leq \frac{1}{2} + \epsilon.$$

D'où $R(n) = O(n^{1/2+\epsilon})$. Dans le cas du report, il faut rendre les ensembles canoniques des noeuds sélectionnés. Pour chaque noeud, cet ensemble est l'union des points stockés aux feuilles du sous-arbre dont il est racine. Comme ce sous-arbre à une taille linéaire, on en déduit que le problème du report peut-être résolu avec un temps supplémentaire proportionnel à la taille de la réponse.

Note : En prenant $r = \sqrt{n}$ et en résolvant $R(n) = O(r) + O(\sqrt{r})R(2n/r)$ on trouve $R(n) = O(\sqrt{n}\, 2^{O(\log\log n)}) = O(\sqrt{n}\, polylog\, n)$

Pour répondre à une requête de type triangle au lieu de demi-plan on peut utiliser la même structure et obtenir la même complexité asymptotique de recherche en remarquant que, à chaque niveau de la recherche, le nombre de triangles d'une partition intersectés par les 3 droites supports d'un triangle de requête est en $O(3\sqrt{r})$. Finalement :

Lemme 7.14 *Le problème de la recherche simpliciale dans le plan peut se résoudre en utilisant un arbre de partitions avec :*
pré-calcul : $O(n \log n)$
espace : $O(n)$
temps requête : $O(n^{1/2+\epsilon})$ *pour le décompte et* $O(n^{1/2+\epsilon} + k)$ *pour le report où* k *est le nombre de points à reporter.*

7.2.2 Arbres de cuttings et recherche par demi-plan

Les arbres de partitions ont évidemment une taille optimale mais un temps de requête important. Quitte à augmenter la taille de la structure de recherche, il est possible de d'obtenir des temps de recherche logarithmiques. C'est l'objet de ce qui suit. La méthode repose ici sur une transformation préalable des données par la dualité point/droite du plan. Celle ci, notée $*$, est donnée par la transformation $(a, b) \mapsto \{y = ax - b\}$, et son inverse, et définit une bijection entre le plan et l'ensemble des droites non verticales du plan.

Propriété : le point p est au dessus de la droite d si et seulement si d^* est au dessus de p^*.

Étant donné une droite d, trouver tous les points s_i de S au dessus de d revient ainsi à trouver l'ensemble des droites de $\{s_i^* | s_i \in S\}$ au dessous de d^*. Clairement cet ensemble ne dépend que de la cellule de l'arrangement des droites s_i^* contenant d^*. Il y a $O(n^2)$ telles cellules. On peut donc espérer un algorithme utilisant une place $O(n^2)$ avec un temps de requête en $O(\log n)$.

Définition 7.15 *Soit L un ensemble de n droites du plan et soit $r \in [1, n]$. Un* $(1/r)$-cutting *pour L est une partition du plan en triangles (possiblement non bornés) telle que chaque triangle est intersecté par au plus n/r droites de L. La* taille *de ce cutting est son nombre de triangles.*

Le théorème 8.1 affirme que l'on peut construire en temps $O(nr)$ un $(1/r)$-cutting de taille $O(r^2)$ en collectant pour chaque triangle du cutting les droites de L qui le coupent.

On s'intéresse tout d'abord au problème du décompte. On définit pour cela récursivement un *arbre de cuttings* pour L :
- si L contient une unique droite alors son arbre est réduit à une feuille qui stocke cette droite,
- sinon on crée autant d'enfants de la racine que de triangles dans un $(1/r)$-cutting de L. Pour chaque enfant ν, associé au triangle $\Delta(\nu)$, on construit récursivement un arbre de cuttings pour les droites $L(\nu) \subset L$ intersectant $\Delta(\nu)$. L'ensemble de droites $L(\nu)$ est appelé l'*ensemble canonique* de ν. Pour le problème du comptage on stocke également pour chaque enfant les nombres $\ell^-(\nu)$ et $\ell^+(\nu)$ de droites de L respectivement au dessus et au dessous de $\Delta(\nu)$.

La taille $T(n)$ d'un arbre de cuttings construit à l'aide de $(1/r)$-cuttings, chacun de taille $O(r^2)$, vérifie

$$T(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(r^2) + \sum_\nu T(n_\nu) & \text{sinon} \end{cases} \tag{7.5}$$

Par définition d'un $1/r$-cutting on a encore pour une certaine constante c :

$$T(n) \leq cr^2 + cr^2 T(n/r)$$

D'où $T(n) = O(n^{\log_r cr^2})$. Soit en prenant $r = \lceil c^{1/\epsilon} \rceil$, $T(n) = O(n^{2+\epsilon})$.

Le temps $C(n)$ de construction de cette structure vérifie

$$C(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(nr) + \sum_\nu C(n_\nu) & \text{sinon} \end{cases} \tag{7.6}$$

Soit encore $C(n) \leq cnr + cr^2 C(n/r)$ pour une certaine constante c, ce qui donne à nouveau $C(n) = O(n^{2+\epsilon})$ pour $r = \lceil c^{1/\epsilon} \rceil$.

Pour trouver le nombre de droites sous un point p de requête on descend récursivement dans l'arbre de cuttings depuis la racine jusqu'aux feuilles en s'orientant pour chaque noeud visité vers l'unique enfant dont le triangle associé contient p. La somme des nombres ℓ^- associés aux noeuds de ce parcours est le nombre cherché. Bien entendu, en accumulant les valeurs ℓ^+ au lieu de ℓ^- la même structure permet de calculer le nombre de droites au dessus d'un point de requête.

Le temps de recherche $R(n)$ vérifie ainsi :

$$R(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(r^2) + R(n/r) & \text{sinon} \end{cases} \tag{7.7}$$

D'où $R(n) = O(\log n)$ si r est constant.

Pour le problème du report, pour chaque noeud ν de l'arbre de cuttings, si μ est son parent, on stocke explicitement au niveau de ν les sous-ensembles $L^+(\nu), L^-(\nu) \subset L(\mu)$ de droites respectivement au dessus et au dessous du triangle $\Delta(\nu)$. L'ensemble des droites au dessous (resp. au dessus) d'un point de requête est obtenu récursivement comme dans le cas du décompte en accumulant cette fois les ensembles L^- (resp. L^+) le long du parcours. Le temps de requête se décompose alors en un temps de parcours $O(\log n)$ comme pour le décompte plus un terme proportionnel à la taille de la réponse. La taille $T'(n)$ de la structure de recherche vérifie maintenant

$$T'(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(nr^2) + \sum_\nu T'(n_\nu) & \text{sinon} \end{cases} \tag{7.8}$$

D'où $T'(n) \leq O(nr^2) + r^2 T'(n/r)$. Par application du master théorème (section 1.9) on déduit à nouveau $T'(n) = O(n^{2+\epsilon})$. En résumé,

Lemme 7.16 *Le problème de la recherche par demi-plan peut se résoudre en utilisant arbre de cuttings avec :*
pré-calcul : $O(n^{2+\epsilon})$
espace : $O(n^{2+\epsilon})$
temps requête : $O(\log n)$ *pour compter et* $O(\log n + k)$ *pour reporter où* k *est le nombre de points à reporter.*

7.2.3 Application à la recherche simpliciale

Soit Δ un triangle, intersection de trois demi-espaces H_1, H_2, H_3 bordés par des droites h_1, h_2, h_3. On suppose que H_1 est au dessous de h_1 tandis que H_2 et H_3 sont au dessus de leur droite bordante respective. Les autres cas se traitent de manière analogue. Le problème de la sélection des points de S inclus dans Δ se dualise alors en un problème du type : trouver les droites de S^* simultanément au dessus de h_1^* et au dessous de h_2^* et de h_3^*.

Pour répondre à ce type de requête – et donc par dualité à une requête de type triangle – on peut utiliser un structure d'arbre de cuttings à trois niveaux : le premier niveau est un arbre de cuttings simple. On associe de plus à chaque noeud ν deux arbres de cuttings (à deux niveaux) : l'un sur son ensemble $L^+(\nu)$ et l'autre sur son ensemble $L^-(\nu)$. Finalement on associe à chaque noeud de la structure secondaire deux arbres de cuttings simples sur ses ensembles L^+ et L^-. La recherche procède comme suit. On effectue une requête avec h_1^* sur l'arbre primaire pour sélectionner les droites au dessus de h_1^*. Pour chaque noeud ν sélectionné dans cette recherche, plutôt que de rendre (ou comptabiliser) $L^+(\nu)$, on effectue une requête avec h_2^* sur sa structure secondaire afin de sélectionner les droites de $L^+(\nu)$ au dessous de h_2^*. Finalement on effectue une requête avec h_3^* sur les structures associées aux noeuds secondaires sélectionnés pour rechercher les droites au dessous de h_3^*. Clairement, cette procédure permet de sélectionner les droites simultanément au dessus de h_1^* et au dessous de h_2^* et de h_3^*.

Le temps de recherche dans une structure secondaire vérifie :

$$R'(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(\log n + r^2) + R'(n/r) & \text{sinon} \end{cases} \tag{7.9}$$

D'où $R'(n) = O(\log^2 n)$ si r est constant. Le temps de recherche dans la structure primaire vérifie donc :

$$R(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(\log n^2 + r^2) + R(n/r) & \text{sinon} \end{cases} \tag{7.10}$$

D'où $R(n) = O(\log^3 n)$ si r est constant. La taille de la structure secondaire vérifie :

$$T'(n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ O(r^2 + n^{2+\epsilon}) + \sum_\nu T'(n_\nu) & \text{sinon} \end{cases} \tag{7.11}$$

Soit (cf. master theorem) $T'(n) = O(n^{2+\epsilon})$ si r est constant et suffisamment grand. On en déduit de même que la taille de la structure primaire est un $O(n^{2+\epsilon})$. On montre selon les mêmes lignes que celle-ci peut-être construite dans le même temps. Finalement,

Lemme 7.17 *Le problème de la recherche simpliciale dans le plan peut se résoudre en utilisant un arbre de cuttings à trois niveaux avec :*
pré-calcul : $O(n^{2+\epsilon})$
espace : $O(n^{2+\epsilon})$
temps requête : $O(\log^3 n)$ *pour compter et* $O(\log^3 n + k)$ *pour reporter où* k *est le nombre de points à reporter.*

Chapitre 8

Cuttings et partitions simpliciales

8.1 Cuttings

Soit L un ensemble de n droites du plan en position générale et soit $r \in [1, n]$. On rappelle qu'un *$(1/r)$-cutting* pour L est un découpage du plan en triangles généralisés (i.e. des intersections de 3 demi-espaces et donc possiblement non bornés) telle que l'intérieur de chaque triangle rencontre au plus n/r droites[1] de L. On entend par découpage que les intérieurs des triangles sont deux à deux disjoints et que leur union couvre le plan. La *taille* de ce cutting est son nombre de triangles.

L'objectif est de construire un $(1/r)$-cutting de la plus petite taille possible. L'argument suivant montre que cette taille est minorée par un $\Omega(r^2)$. Par l'hypothèse de position générale, l'arrangement des n droites de L possède $\binom{n}{2} = \Omega(n^2)$ sommets. Un triangle d'un $(1/r)$-cutting étant coupé par au plus n/r droites, il contient au plus $\binom{n/r}{2} = O((n/r)^2)$ de ces sommets. Le plan doit donc être couvert par au moins $\Omega(r^2)$ triangles. Le théorème suivant montre que l'on peut construire des $(1/r)$-cuttings dont la taille est optimale à un facteur constant près. La preuve repose sur une analyse d'échantillonnages aléatoires *à la* Clarkson [CS89, CMS93]. Une approche non randomisée due à B. Chazelle et J. Friedman [CF90] est exposée par J. Matoušek dans [Mat91]. Voir aussi [PA95, p. 173-175].

Théorème 8.1 *Pour tout ensemble L de n droites du plan et tout $r \leq n$ il existe un $(1/r)$-cutting de taille $O(r^2)$ qui peut être construit en temps $O(nr)$ en collectant de plus pour chaque triangle les droites qui le coupent.*

La construction repose sur une triangulation de l'arrangement d'un sous-ensemble des droites de L. Cette triangulation peut être obtenue de manière canonique en triangulant chaque cellule (polygone) de l'arrangement à partir de son sommet le plus bas (minimal pour l'ordre lexicographique sur les coordonnées), c'est à dire en remplaçant chaque cellule par le cône issu du sommet le plus bas sur les arêtes du polygone ne contenant

[1] Pour étendre le résultat à des arrangements non-simples on voit bien qu'il est nécessaire de restreindre la condition à l'intérieur des triangles puisque tout sommet de degré $2n/r$ ou plus de l'arrangement de L est incident à au moins n/r droites.

pas ce sommet. De manière générale on obtient la *triangulation canonique* d'un arrangement d'hyperplans de $\mathbb{R}^d$ en triangulant récursivement le i-squelette de l'arrangement par "étoilement" de chaque i-cellule à partir de son sommet le plus bas.[2]

Montrons tout d'abord comment collecter les droites coupant chaque triangle d'un $(1/r)$-cutting de taille $k = O(r^2)$. Remarquons tout d'abord que l'on peut déterminer pour chaque droite $\ell \in L$ un triangle la coupant en temps $O(\log k)$. Il suffit par exemple de pré-calculer pour une droite donnée la liste ordonnée de ses intersections avec les triangles du cutting, puis de situer en temps $O(\log k)$ dans cette liste l'intersection de ℓ avec cette droite. Une fois connu un triangle intersectant ℓ il suffit de longer ℓ dans le cutting (donné sous forme de liste d'adjacence entre triangles[3]) pour déterminer tous les triangles du cutting coupés par ℓ. Ceci prend un temps proportionnel au nombre n_ℓ de triangles coupés. Le temps total requis pour collecter toutes les intersections est alors $\sum O(\log k + n_d)$, la somme portant sur les n droites de L. Or $\sum n_d \leq k(n/r)$ par définition d'un $(1/r)$-cutting. Pour $k = O(r^2)$ comme dans le théorème,on obtient un temps $O(nr)$.

On montre d'abord une version sous-optimale du théorème.

Lemme 8.2 *Pour tout ensemble L de n droites du plan et tout $r \leq n$ il existe un $(1/r)$-cutting de taille $O(r^2 \log^2 r)$ qui peut être construit en temps $O(r^2 \log^2 r + nr \log r)$ en collectant de plus pour chaque triangle les droites qui le coupent.*

Preuve I : Considérons l'ensemble des droites coupant l'intérieur d'un triangle quelconque. Lorsque le triangle décrit tous les triangles (généralisés) possibles du plan, les ensembles de droites correspondant forment un système de parties (cf. Chapitre **??**). On montre que ce système a une dimension de Vapnik-Chervonenkis finie (cf. [PA95, p.256-257]). Par le corollaire **??**, ce système admet un $\frac{1}{r}$-net R de taille $O(r \log r)$. La triangulation canonique $\mathcal{T}(R)$ de l'arrangement des droites de R comporte donc $O(r^2 \log^2 r)$ triangles. De plus, par définition d'un $\frac{1}{r}$-net, un triangle dont l'intérieur est disjoint de R – c'est en particulier le cas des triangles de $\mathcal{T}(R)$ – est coupé par au plus n/r droites de L. Les triangles de $\mathcal{T}(R)$ forment donc un $(1/r)$-cutting de taille $O(r^2 \log^2 r)$. □

La construction d'un ϵ-net de petite taille peut s'obtenir par échantillonnage aléatoire. La preuve suivante s'appuie directement sur un échantillonnage aléatoire de L sans passer par la théorie de Vapnik-Chervonenkis. Elle utilise en contrepartie quelques résultats sur les échantillonnages aléatoires explicités plus loin.

Preuve II : Soit R un échantillon aléatoire de taille $t = cr \log r$ pour la loi uniforme sur $\binom{L}{t}$ (cf. section 1.8.5) où c est une constante déterminée plus loin. Le théorème 8.4 ci-après indique qu'avec une forte probabilité un triangle de la triangulation canonique $\mathcal{T}(R)$ de l'arrangement de R intersecte au plus $b\frac{n}{t} \log t$ droites de L pour une certaine

[2] De manière équivalente on peut trianguler récursivement le $(d-1)$-arrangement induit dans chaque hyperplan par les autres hyperplans de l'arrangement puis étoiler les d-cellules à partir de leur sommet le plus bas. Cette méthode fournit bien la même triangulation si on prend garde de déduire correctement le système de coordonnées de chaque hyperplan par "projection" du système de coordonnées canonique.

[3] On suppose implicitement ici que le cutting est formé à partir de la triangulation canonique d'un arrangement, de sorte que chaque triangle a au plus trois triangles adjacents.

constante b. En particulier, $\mathcal{T}(R)$ est un $b\log t/t$-cutting de taille $O(t^2) = O(r^2\log^2 r)$. Or

$$b\log t/t = b\frac{\log(cr\log r)}{cr\log r} = \frac{b}{rc}(1+\frac{\log\log r}{\log r}+\frac{\log c}{\log r})$$

cette dernière quantité étant majoré par $1/r$ dès que $r \geq c \geq 3b$. Pour $r < c$ il suffit de choisir $t = c^2\log c < Kr\log r$ pour une certaine constante K puisqu'un $\frac{1}{c}$-cutting est alors un $\frac{1}{r}$-cutting. □

Preuve du théorème 8.1 : On considère un r-échantillon aléatoire R pour la loi uniforme sur $\binom{L}{r}$. Certains triangles de la triangulation canonique $\mathcal{T}(R)$ de l'arrangement de R sont possiblement coupés par plus de n/r droites de L. L'idée est de subdiviser chaque triangle Δ de $\mathcal{T}(R)$ par un cutting restreint au sous-ensemble de droites $L_\Delta \subset L$ coupant ce triangle. On peut en effet s'arranger pour que les triangles de la subdivision soient coupés par au plus n/r droites tout en utilisant $O(r^2)$ triangles au total dans les subdivisions. Pour cela on construit à l'aide du lemme 8.2 pour chaque triangle Δ un $1/r_\Delta$-cutting $\mathcal{C}_\Delta$ de L_Δ composé de $O(r_\Delta^2\log r_\Delta^2)$ triangles en choisissant $r_\Delta = n_\Delta r/n$ où $n_\Delta = |L_\Delta|$ (bien sûr, on ne fait rien si $n_\Delta \leq n/r$). Chaque triangle de ce cutting est coupé par au plus $n_\Delta/r_\Delta = n/r$ droites. Ceci est encore vrai pour les $O(r_\Delta^2\log r_\Delta^2)$ triangles obtenus en intersectant $\mathcal{C}_\Delta$ avec Δ et en retriangulant les éventuels k-gones ($k \leq 6$) résultant. La réunion de ces subdivisions est donc un $1/r$-cutting $\mathcal{C}$ de L de taille $O(r^2) + \sum_{\Delta\in\mathcal{T}(R)} r_\Delta^2\log r_\Delta^2$. Évaluons l'espérance de cette dernière somme relativement à R.

$$E(\sum_\Delta r_\Delta^2\log r_\Delta^2) \leq E(\sum_\Delta r_\Delta^4) = (r/n)^4 E(\sum_\Delta n_\Delta^4)$$

Par le lemme 8.7, $E(\sum_\Delta n_\Delta^4) = O((n/r)^4 r^2)$. On en déduit que $\mathcal{C}$ a en moyenne $O(r^2)$ triangles. □

On pourra consulter [HP00] pour des constructions effectives de cuttings avec des bornes effectives sur la complexité de l'algorithme.

Le théorème 8.1 admet une version pondérée. Cette fois on se donne une famille pondérée (L, w) de n droites avec des poids $w = (w_1, w_2, \ldots, w_n)$ non-négatifs. Un $1/r$-cutting est alors tel que le poids total des droites de L coupant l'intérieur d'un triangle du cutting est majoré par n/r.

Corollaire 8.3 *Pour toute famille pondérée (L, w) de n droites du plan et tout $r \leq n$ on peut construire en temps $O(nr)$ un $(1/r)$-cutting de taille $O(r^2)$ en collectant de plus pour chaque triangle les droites qui le coupent.*

Preuve : Quitte à renormaliser les poids en temps linéaire, on peut supposer que $\sum_i w_i = 1$. On considère le multi-ensemble L' de droites obtenu en incluant chaque droite de L de poids w_i avec multiplicité $\lceil w_i\rceil n$. Notons que $|L'| \leq 2n$. Suivant le théorème 8.1, on calcule en temps $O(nr)$ un $\frac{1}{2r}$-cutting de taille $O(r^2)$ pour la collection non-pondérée de droites L'. Pour cela on peut considérer que toutes les droites de L' sont distinctes à l'aide de perturbations symboliques. Puisque chaque triangle est coupé par au plus $|L'|/(2r) \leq n/r$ droites de L', ce cutting est un $(1/r)$-cutting pour la famille pondérée (L, w). □

8.2 Échantillonnage aléatoire

On effectue ici quelques calculs en moyenne portant sur des arrangements de droites et utiles pour le calcul de cuttings. On s'en tient au cadre des arrangements de droites et de leur triangulation canonique (cf. section 8.1) mais les résultats peuvent être établis dans un cadre plus abstrait comme au chapitre 9. Le terme d'échantillon aléatoire se réfère toujours à la loi uniforme (cf. section 1.8.5) mais les résultats ci-dessous peuvent être établis avec d'autres lois. On pourra consulter [Mul00] à ce sujet.

Soit L un ensemble de n droites du plan en position générale. On note $\mathcal{T}^0(L)$ l'ensemble des triangles de la triangulation canonique de L. On note ensuite $\mathcal{T}(L)$ l'ensemble des triangles *réalisables sur* L c.-à-.d apparaissant dans la triangulation canonique d'au moins une partie de L, i.e. $\mathcal{T}(L) = \bigcup_{R\subset L} \mathcal{T}^0(R)$. Pour tout triangle $\sigma \in \mathcal{T}(L)$ on note L_σ l'ensemble des droites de L qui rencontrent l'intérieur de σ et on pose $\ell_\sigma := |L_\sigma|$. On note également D_σ l'ensemble des droites de $L \setminus L_\sigma$ qui rencontrent le bord de σ. On dit que D_σ *détermine* σ. Notons que par l'hypothèse de position générale le *degré* $d_\sigma := |D_\sigma|$ de σ est majoré par 5. De plus σ est dans la triangulation canonique d'un échantillon $R \subset L$ si et seulement si $D_\sigma \subset R \subset L \setminus L_\sigma$. En particulier, le nombre de triangles possédant le même ensemble déterminant est majoré par une constante (la taille maximale de la triangulation canonique de 5 droites). On en déduit

$$\mathcal{T}(L) = O(n^5)$$

On note enfin $\mathcal{T}^i(R)$ l'ensemble des triangles σ réalisables sur l'échantillon R tels que $|L_\sigma \cap R| = i$.

Théorème 8.4 *Soit R un r-échantillon aléatoire pour la loi uniforme sur $\binom{L}{r}$ alors pour tout $\epsilon > 0$*

$$P\big(\max_{\sigma\in\mathcal{T}^0(R)} \ell_\sigma \le c_\epsilon \frac{n}{r}\log r\big) > 1-\epsilon$$

pour une certaine constante c_ϵ. Dit autrement, avec une forte probabilité chacun des triangles de la triangulation canonique de R est coupé (en son intérieur) par au plus $O(\frac{n}{r}\log r)$ droites de L.

Preuve : On note $q(c)$ la probabilité pour qu'il existe un triangle de la triangulation canonique de R qui soit coupé par plus de $c\frac{n}{r}\log r$ droites de L. Il suffit donc de montrer que l'on peut choisir c de sorte que $q(c) \le \epsilon$. On a

$$q(c) = P\Big(\bigvee_{\substack{\sigma\in\mathcal{T}(L)\\ \ell_\sigma > c\frac{n}{r}\log r}} \sigma \in \mathcal{T}^0(R)\Big) \le \sum_{\substack{\sigma\in\mathcal{T}(L)\\ \ell_\sigma > c\frac{n}{r}\log r}} P\big(\sigma\in\mathcal{T}^0(R)\big)$$

Puisque $\mathcal{T}^0(R) \subset \mathcal{T}(R)$ on peut écrire

$$P\big(\sigma\in\mathcal{T}^0(R)\big) = P\big(\sigma\in\mathcal{T}^0(R) \mid \sigma\in\mathcal{T}(R)\big)\cdot P\big(\sigma\in\mathcal{T}(R)\big)$$

Or

$$P\big(\sigma\in\mathcal{T}^0(R) \mid \sigma\in\mathcal{T}(R)\big) = \binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma}\Big/\binom{n-d_\sigma}{r-d_\sigma} \le (1-\frac{\ell_\sigma}{n-d_\sigma})^{r-d_\sigma} \le e^{-\frac{\ell_\sigma}{n-d_\sigma}(r-d_\sigma)}$$

Avec les hypothèses $\ell_\sigma > c\frac{n}{r}\log r$ et $d_\sigma \leq 5$ on en déduit $P(\sigma \in \mathcal{T}^0(R) \mid \sigma \in \mathcal{T}(R)) < r^{-\frac{c}{\ln 2}(1-\frac{5}{r})}$. D'où

$$q(c) \leq r^{-\frac{c}{\ln 2}(1-\frac{5}{r})} \sum_{\substack{\sigma \in \mathcal{T}(L) \\ \ell_\sigma > c\frac{n}{r}\log r}} P(\sigma \in \mathcal{T}(R)) \leq r^{-\frac{c}{\ln 2}(1-\frac{5}{r})} E(|\mathcal{T}(R)|)$$

Mais comme noté plus haut le nombre $|\mathcal{T}(R)|$ de triangles réalisables sur R est un $O(r^5)$. D'où

$$q(c) \leq r^{5-\frac{c}{\ln 2}(1-\frac{5}{r})}$$

On peut donc choisir c (indépendamment de r) pour rendre cette quantité aussi petite que désirée (notons que pour r petit, disons $r \leq 10$, le théorème est trivialement vrai). □

Lemme 8.5 *Soit R un r-échantillon de L et soit Q un $r/2$-échantillon aléatoire de R. Alors pour tout i*

$$|\mathcal{T}^i(R)| \leq c_i E(|\mathcal{T}^0(Q)|) = O(r^2)$$

pour une certaine constante $c_i > 0$.

Preuve : C'est évident si $\mathcal{T}^i(R)$ est vide. On pose pour tout $\sigma \in \mathcal{T}^i(R)$,

$$p(\sigma) := P(\sigma \in \mathcal{T}^0(Q)) = P(D_\sigma \subset Q \subset R \setminus L_\sigma)$$

les probabilités étant relatives à la loi uniforme sur $\binom{R}{r/2}$. Puisque $|R \cap L_\sigma| = i$, on a

$$p(\sigma) = \binom{r - d_\sigma - i}{r - d_\sigma} \Big/ \binom{r}{r/2}$$

On montre (cf. exercice ci-dessous) que cette dernière quantité est minorée par une constante $k_i > 0$ dès que $r \geq \max(16, 4i-4)$. On a alors

$$E(|\mathcal{T}^0(Q)|) = \sum_{\sigma \in \mathcal{T}(R)} p(\sigma) \geq \sum_{\sigma \in \mathcal{T}^i(R)} p(\sigma) \geq k_i |\mathcal{T}^i(R)|$$

Ce qui permet de conclure puisque $|\mathcal{T}^0(Q)| = O(r^2)$ d'après le théorème 5.3 sur la complexité des arrangements. Notons que quitte à augmenter c_i, l'inégalité du lemme est inconditionnellement vraie puisque pour $i < r < \max(16, 4i-4)$, $E(|\mathcal{T}^0(Q)|)$ est uniformément minorée par une constante positive tandis que $|\mathcal{T}^i(R)|$ est uniformément majoré. □

Exercice 8.6 *Montrer que pour $r \geq \max(4d-4, 4i-4)$:*

$$\binom{r - d - i}{r - d} \Big/ \binom{r}{r/2} \geq 1/4^{i+d}$$

Lemme 8.7 *Pour tout i on a la majoration suivante pour l'espérance du moment d'ordre i du nombre de droites coupant un triangle de la triangulation canonique d'un r-échantillon aléatoire R (pour la loi uniforme) :*

$$E(\sum_{\sigma\in\mathcal{T}^0(R)}\binom{\ell_\sigma}{i}) = O((n/r)^i r^2)$$

Preuve : Pour tout triangle σ réalisable sur L et pour $j = 0$ ou i on pose $p_j(\sigma) = P(\sigma \in \mathcal{T}^j(R))$. On a ainsi

$$p_0(\sigma) = P(D_\sigma \subset R \subset L \setminus L_\sigma) = \binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma}/\binom{n}{r}$$

et

$$p_i(\sigma) = P(D_\sigma \subset R \text{ et } |L_\sigma \cap R| = i) = \binom{\ell_\sigma}{i}\binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma-i}/\binom{n}{r}$$

D'où $\binom{\ell_\sigma}{i}p_0(\sigma) = p_i(\sigma)\binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma}/\binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma-i}$.

Or dès que $r > 2(i+4)$ on a (cf. exercice ci-dessous) $\binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma}/\binom{n-d_\sigma-\ell_\sigma}{r-d_\sigma-i} < c_i(n/r)^i$ pour une certaine constante c_i. On en déduit

$$E(\sum_{\sigma\in\mathcal{T}^0(R)}\binom{\ell_\sigma}{i}) = \sum_{\sigma\in\mathcal{T}(L)}\binom{\ell_\sigma}{i}p_0(\sigma) \le c_i(n/r)^i\sum_{\sigma\in\mathcal{T}(L)}p_i(\sigma) \le c_i(n/r)^i E(|\mathcal{T}^i(R)|)$$

Et on conclut avec le lemme précédent. □

Exercice 8.8 *Montrer que pour $r > 2(i+d-1)$ on a*

$$\binom{n-d-\ell}{r-d}/\binom{n-d-\ell}{r-d-i} < 2^i(n/r)^i$$

Note : Une preuve alternative du théorème 8.4 utilise la notion d'ϵ-net (cf. définition **??**). En voici une esquisse. On se référera au chapitre **??** pour les définitions et résultats appropriés. On considère le système de parties $(\mathcal{D}, \{\Delta_t\}_t)$ où $\mathcal{D}$ est l'ensemble des droites du plan et où $\{\Delta_t\}_t$ est l'ensemble des parties de $\mathcal{D}$ indexé par les triangles du plan avec Δ_t défini comme le sous-ensemble des droites rencontrant l'intérieur du triangle t. On montre (cf. proposition **??**) que ce système a une dimension de Vapnik-Chervonenkis finie. On en déduit par le corollaire **??** que tout ensemble L de n droites possède un $\frac{\log r}{r}$-net E de taille $O(r)$ relativement à $(\mathcal{D}, \{\Delta_t\}_t)$. Dit autrement, tout triangle du plan coupé par au moins $\frac{n}{r}\log r$ droites de L contient une droite de E. Par conséquent, les triangles de la triangulation canonique de E sont coupés par au plus $\frac{n}{r}\log r$ droites de L. Le théorème **??** montre finalement que E peut être obtenu par tirage aléatoire avec une bonne probabilité.

8.3 Partitions Simpliciales

On rappelle qu'une r-partition (simpliciale) d'un ensemble S de n points du plan est une famille de couples $\{(S_i, \Delta_i)\}_{i\in I}$ où les S_i forment une partition de S et ont chacun une taille comprise entre n/r et $2n/r$ et où Δ_i est un triangle du plan contenant S_i. Le triangle Δ_i peut éventuellement être dégénéré en un segment, ce qui est utile lorsque S n'est pas en position générale. Notons que la *taille* $|I|$ de la partition est majorée par r. Son *nombre de croisements* relativement à un ensemble de droites et le nombre maximal de triangles intersectés (transversalement pour les triangles dégénérés) par l'une de ces droites. On sous-entendra l'ensemble de toutes les droites du plan lorsque cet ensemble n'est pas spécifié. Je suis la présentation de Matoušek [Mat92].

Théorème 8.9 (de la partition simpliciale, Matoušek 1992) *Pour tout ensemble S de n points et pour tout r ($2 \le r \le n/2$), il existe une r-partition de S de nombre de croisements $O(\sqrt{r})$ qui peut être construite en temps $O(n)$ si r est majoré par une constante.*

La preuve procède en deux étapes. Dans un premier temps on montre que pour tout ensemble de droites fini L, on peut construire une r-partition de S de nombre de croisements $O(\sqrt{r} + \log|L|)$ relativement à L. On montre ensuite qu'on peut construire un ensemble test L_0 composé de $O(r)$ droites tel que le nombre de croisements de *toute* r-partition de S est en gros égal à son nombre de croisements relativement à L_0.

Lemme 8.10 *Soit S, n, r comme dans le théorème et soit L un ensemble fini de droites. On peut construire en temps $O(nr\log r + |L|r^{3/2})$ une r-partition de S de nombre de croisements $O(\sqrt{r} + \log|L|)$ relativement à L.*

Preuve : On considère la famille de droites pondérées (L, w_1) avec des poids unitaires ; on note $|w_1| = |L|$ son poids total. On pose $\Sigma_1 = S$ et $r_1 = r$. Par le corollaire 8.3 on peut construire un $\frac{1}{c\sqrt{r_1}}$-cutting de (L, w_1), pour une constante c appropriée, possédant au plus r_1 faces (triangles et arêtes) au total. En particulier, le poids total des droites de L coupant une triangle de ce cutting est majoré par $O(|w_1|/\sqrt{r_1})$. Il en est de même pour les arêtes, puisque toute droite qui coupe une arête coupe ses triangles incidents. L'une des faces Δ_1 du cutting contient au moins $|\Sigma_1|/r_1 = n/r$ sommets de Σ_1. On choisit arbitrairement un sous-ensemble S_1 de $\lceil n/r\rceil$ sommets dans Δ_1, ce qui fournit la première paire (S_1, Δ_1) de la r-partition. On pose ensuite $\Sigma_2 = \Sigma_1 \setminus S_1$, $r_2 = |\Sigma_2|r/n$ et on définit des poids w_2 pour L en doublant les poids des droites de L coupant Δ_1 et en laissant inchangé les autres poids. Si $|\Sigma_2| \le 2n/r$, on pose $S_2 = \Sigma_2$ et $\Delta_2 = \mathbb{R}^2$. Sinon on itère le procédé avec Σ_2, r_2 et (L, w_2). Au bout de $m \le r$ étapes, on obtient une r-partition simpliciale de S :

$$(S_1, \Delta_1), (S_2, \Delta_2), \ldots, (S_m, \Delta_m)$$

On considère une droite $\ell \in L$. Soit k le nombre de triangles de la r-partition coupés par ℓ. On a ainsi $w_{m+1}(\ell) = 2^k$. Par ailleurs, si p est le poids total des droites de L coupant Δ_i à l'étape i on a d'une part $p = O(|w_i|/\sqrt{r_i})$, par les propriétés du cutting

utilisé, et d'autre part $|w_{i+1}| = (|w_i| - p) + 2p$, par la règle de doublement des poids. D'où $|w_{i+1}| = |w_i|(1 + O(\frac{1}{\sqrt{r_i}}))$, et partant de $|w_1| = |L|$:

$$|w_{m+1}| = |w_1| \prod_{i=1}^{m} (1 + O(\frac{1}{\sqrt{r_i}})) \leq |L| e^{O(\sum_{i=1}^{m} \frac{1}{\sqrt{r_i}})}$$

Cette dernière relation provenant de l'inégalité $1 + x \leq e^x$. Or,

$$r_i = |\Sigma_i| r/n = (n - (i-1)\lceil n/r \rceil) r/n \geq r - i + 1$$

Il suit que $\sum_{i=1}^{m} \frac{1}{\sqrt{r_i}} \leq \sum_{i=1}^{r} \frac{1}{\sqrt{i}} \leq 2\sqrt{r}$ en majorant la somme par une intégrale. On déduit finalement de $w_{m+1}(\ell) \leq |w_{m+1}|$ la majoration cherchée : $k = O(\log |L| + \sqrt{r})$.

Le temps requis à chaque étape est $O(|L|\sqrt{r_i})$ pour la construction du cutting plus $O(n_i \log r_i)$ pour sélectionner les points de S_i en utilisant par exemple une structure de recherche pour le cutting telle que décrite section 6.2. Le temps total de construction de la r-partition est donc $O(|L| r^{3/2} + nr \log r)$. □

Lemme 8.11 (de l'ensemble test) *Soient S, n, r comme dans le théorème. Il existe un ensemble L de $O(r)$ droites tel que le nombre de croisements de toute r-partition $\mathcal{P}$ de S est majoré par*

$$3c_L + \sqrt{r}$$

où c_L est le nombre de croisements de $\mathcal{P}$ relativement à L. De plus, un tel ensemble L peut être construit en temps $O(n\sqrt{r})$.

Preuve : On note $*$ la dualité point/droite comme à la section 7.2.2. Soit S^* l'ensemble des droites duales des points de S. Par le théorème 8.1, on peut construire en temps $O(n\sqrt{r})$ un $\frac{1}{\sqrt{r}}$-cutting de S^* constitué de $O(r)$ triangles, arêtes et sommets. On définit L comme les droites duales des $O(r)$ sommets de ce cutting. Reste à vérifier que L a la propriété désirée. Soient $\mathcal{P}$ et c_L comme dans le lemme et soit ℓ une droite du plan. On considère le triangle Δ du cutting contenant le point dual ℓ^* et on note D l'ensemble des droites duales des (au plus 3) sommets de Δ. Soit c le nombre de triangles de $\mathcal{P}$ coupés par ℓ et par au moins l'une des droites de D. Par définition de c_L, on a $c \leq 3c_L$. On note c' le nombre de triangles de $\mathcal{P}$ coupés par ℓ mais par aucune des droites de D. Ces triangles contiennent au moins $c'n/r$ sommets de S et sont contenus dans la zone de ℓ dans l'arrangement des droites de D. Il suit aisément que les droites duales de ces sommets coupent l'intérieur de Δ.[4] Par le choix du cutting, leur nombre est majoré par $n/\sqrt{r}$. D'où $c' \leq \sqrt{r}$. □

Preuve du théorème 8.9 : Par le lemme 8.11, on construit en temps $O(n\sqrt{r})$ un ensemble L de $O(r)$ droites tests pour les r-partitions de S. Par le lemme 8.10, on construit en temps $O(nr \log r + r^{5/2})$ une r-partition de S de nombre de croisements $O(\sqrt{r})$ relativement à L et qui a donc un nombre de croisements du même ordre relativement à toutes les droites du plan d'après le lemme 8.11. □

[4] Considérer pour cela un chemin reliant un tel sommet à un point de ℓ sans couper les droites de D ; l'extrémité sur ℓ de ce chemin se dualise en une droite passant par ℓ^* et coupe donc Δ. Il en est de même pour les droites duales des autres points de ce chemin.

Remarquons que pour r non constant, la preuve du théorème fournit une construction d'une r-partition en temps $O(nr \log r + r^{5/2})$. Il est possible, quitte à augmenter légèrement le nombre de croisements (en $O(r^{1/2+\varepsilon})$), de réduire ce temps à $O(n \log r)$ (cf. [Mat92]). L'idée est de fixer une constante r_0, puis de calculer une r_0-partition $\{(S_i, \Delta_i)\}_i$ pour S, puis une r_0^2-partition pour chaque S_i, et de continuer récursivement $\log_{r_0} r$ fois.

Chapitre 9

Algorithmes randomisés incrémentaux

9.1 Introduction

Étant donnés un ensemble X d'objets (ex : des points du plan, des droites du plan,...), on s'intéresse à la construction d'une certaine structure géométrique $\mathcal{S}(X)$ associée à cet ensemble (ex : l'enveloppe convexe des points, l'arrangement des droites,...). Le principe d'un *algorithme incrémental* est de construire $\mathcal{S}(X)$ récursivement en ajoutant les objets un à un. Plus précisément, si on ordonne les éléments $x_1, \ldots, x_n$ de X, on construit récursivement $\mathcal{S}_i = \mathcal{S}(\{x_1, \ldots, x_i\})$ à partir de $\mathcal{S}_{i-1}$ en insérant x_i dans la structure. Un algorithme incrémental est dit *randomisé* lorsque l'ordre sur les éléments de X est aléatoire. On peut alors considérer l'algorithme comme une variable aléatoire sur l'espace des permutations de X et estimer l'espérance de son coût en temps ou espace. Un tel algorithme est dit de type *Las Vegas* au sens où l'exécution est aléatoire mais pas le résultat final du calcul. La randomisation permet généralement d'obtenir des algorithmes plus simples que dans le cas déterministe, tout en conservant des complexités en moyenne aussi faibles (voire plus faibles) que les complexités dans le cas le pire des algorithmes déterministes. Le modèle probabiliste, c'est à dire le choix de la probabilité sur les permutations de X, est en général le modèle uniforme. On s'en tiendra ici à ce modèle. En particulier, l'*ensemble* $\{x_1, \ldots, x_i\}$ des i premiers objets constitue un i-échantillon aléatoire de X dont la probabilité de réalisation est $1/\binom{n}{i}$. Quicksort est l'archétype des algorithmes randomisés incrémentaux.

9.2 Le formalisme

Il se trouve que la plupart des structures classiques rencontrées en géométrie algorithmique, telles que les enveloppes convexes de points, les arrangements d'hyperplans, les diagrammes de Voronoi, les décompositions trapézoïdales de segments du plan, etc... peuvent être construites selon des algorithmes randomisés incrémentaux. Ses algorithmes possèdent tous les mêmes caractéristiques et peuvent être analysés de manière analogue. Un formalisme abstrait a ainsi pu être développé pour les algorithmes randomisés incrémentaux. Ce formalisme, introduit par Clarkson et Shor [CS89], s'applique aux différentes

structures citées plus haut et évite des analyses spécifiques à chaque structure particulière.

De manière générale, on considère un univers d'*objets* et un ensemble de *configurations*. Par exemple, pour le problème de l'enveloppe convexe de points de $\mathbb{R}^d$, les objets sont les points de $\mathbb{R}^d$ et les configurations correspondent intuitivement (cf. note ci-après) aux demi-espaces de $\mathbb{R}^d$. À toute configuration on associe un sous-ensemble d'objets et on dit que la configuration est *déterminée* par ce sous-ensemble. Dans l'exemple qui précède un sous-ensemble de points détermine un demi-espace si son enveloppe affine est précisément l'hyperplan qui borde ce demi-espace. Il faut ainsi noter qu'une configuration n'est pas à proprement parler un demi-espace mais plutôt la conjonction d'un ensemble de points dont l'enveloppe affine est un hyperplan et d'une orientation qui distingue l'un des deux demi-espaces bordés par cet hyperplan. Une configuration admet également un certain sous-ensemble d'objets, disjoint de l'ensemble qui la détermine, en *conflit* avec elle. Toujours pour le même exemple, un point est en conflit avec tout "demi-espace" qui le contient en son intérieur.

Étant donné un sous-ensemble fini X d'objets, on souhaite construire une certaine structure $\mathcal{S}(X)$ formée de l'ensemble des configurations déterminées par des sous-ensembles de X et sans conflit avec les objets de X. On note $\mathcal{C}^0(X)$ cet ensemble de configurations. Dans le cas du problème de l'enveloppe convexe, X est un ensemble de points de $\mathbb{R}^d$ et $\mathcal{S}(X)$ est l'enveloppe convexe de ces points. Cette enveloppe est constituée d'un ensemble de facettes dont les hyperplans supports sont précisément les bords des demi-espaces s'appuyant sur des points de X et ne contenant aucun point en leur intérieur, c'est à dire des demi-espaces de $\mathcal{C}^0(X)$. On voit ainsi que, sous une hypothèse de position générale sur les objets de X, les éléments de la structure $\mathcal{S}(X)$ (ici les facettes) sont en bijection avec les configurations de $\mathcal{C}^0(X)$. Généralement, la structure $\mathcal{S}(X)$ que l'on souhaite construire est un peu plus riche que la simple collection des configurations de $\mathcal{C}^0(X)$. Dans notre exemple, on peut également vouloir calculer les adjacences entre facettes. Dans la plupart des cas rencontrés $\mathcal{S}(X)$ est simplement un graphe construit sur $\mathcal{C}^0(X)$.

Il est à noter que le formalisme exact des espaces de configurations n'est pas totalement fixé dans la littérature. On pourra comparer Clarkson [CS89], Mulmuley [Mul00] et Matoušek [SU00, chap. 13]. En particulier, la terminologie précédente varie d'un auteur à l'autre. Ainsi, une configuration est parfois appelée *région* ou *domaine*. L'ensemble déterminant une configuration est aussi son ensemble de *définition* ou d'*adjacences* (ou encore de 'triggers' en anglais) tandis qu'un objet en conflit avec une configuration est dit *tuer* ou *stopper* cette configuration. Les configurations déterminées par des sous-ensembles de X sont dites *réalisables* sur X. Les configurations réalisables et sans conflit sur X, i.e. de $\mathcal{C}^0(X)$, sont dites *actives* sur X.

Voici d'autres exemples importants où le formalisme précédent s'applique.

Exemple 9.1 (Arrangement d'hyperplans) *Les objets sont les hyperplans de* $\mathbb{R}^d$. *Un arrangement d'une famille d'hyperplans est constitué des cellules convexes de la subdivision de* $\mathbb{R}^d$ *induite par les hyperplans de cette famille. On serait donc tenté de définir une configuration comme une cellule convexe déterminée par les hyperplans qui la borde. Mais ce nombre d'hyperplans n'est à priori pas borné. Or, pour des raisons d'efficacité algorithmique, il est important que les configurations réalisables soient déterminées par un nombre borné d'objets. Pour remédier à ce problème on enrichit l'arrangement en*

faisant sa triangulation. Ceci peut être fait de manière canonique en triangulant récursivement les faces de l'arrangement par ordre croissant sur leur dimension ; chaque face est triangulée par étoilement à partir de son sommet de coordonnées minimales pour l'ordre lexicographique[1]*. Ceci amène à définir une configuration comme un d-simplexe. Plus précisément, une configuration est un couple (σ, H) formé d'un d-simplexe σ et d'un ensemble H d'hyperplans incidents à σ (en fait à son bord) tels que σ est dans la triangulation canonique de l'arrangement de H. Une telle configuration est dite déterminée par H et en conflit avec les hyperplans qui rencontrent l'intérieur de σ. On voit maintenant que les d-simplexes de la triangulation canonique d'un arrangement d'hyperplans en position générale sont en bijection avec les configurations actives de l'ensemble des hyperplans de l'arrangement. L'hypothèse de position générale est aussi importante pour borner la taille des ensembles déterminant les configurations réalisables.*

Exemple 9.2 (Décomposition trapézoïdale de segments du plan) *Les objets sont les segments du plan. La structure est la décomposition trapézoïdale obtenue par cloisonnement vertical. Une configuration est un couple (τ, S) où τ est un trapèze du plan et S un ensemble de segments incidents à τ et tel que τ soit l'un des trapèzes de la décomposition trapézoïdale de S. Cette configuration est déterminée par S et en conflit avec les segments qui intersectent l'intérieur de τ. Sous une hypothèse simple de position générale sur une famille de segments, les trapèzes de la décomposition trapézoïdale de cette famille sont en bijection avec les configurations actives sur cette famille.*

Exemple 9.3 (Intersection de demi-espaces) *Les objets sont les demi-espaces de $\mathbb{R}^d$. Une configuration est un couple (p, D) formé d'un point p de $\mathbb{R}^d$ et d'un ensemble de demi-espaces dont l'intersection des hyperplans bordant est p. Une telle configuration est dite déterminée par D et en conflit avec tout demi-espace qui ne contient pas p. Sous une hypothèse de position générale, les sommets du polytope intersection d'une famille de demi-espaces sont en bijection avec les configurations actives sur cette famille.*

Exemple 9.4 (Diagramme de Voronoi du plan) *Les objets sont les points du plan (appelés sites). Pour des raisons analogues à l'emploi d'une triangulation canonique d'un arrangement d'hyperplans (cf. exemple 9.1) on considère la triangulation radiale du diagramme de Voronoi de tout sous-ensemble fini de sites. Elle est obtenue en étoilant chaque cellule du diagramme par rapport à son site. Ce qui amène à définir une configuration comme une paire (t, S) où t est un triangle et S un ensemble de sites tels que t apparaît dans la triangulation radiale du diagramme de Voronoi de S et que toute cellule de ce diagramme rencontre (la clôture de) t. Une telle configuration est dite déterminée par S et en conflit avec tout point p tel que t ne soit pas un triangle de la triangulation du diagramme de Voronoi de $S \cup \{p\}$, i.e. tel qu'il existe un point de t qui soit plus proche de p que des points de S.*

Exemple 9.5 (Diagramme de Delaunay du plan) *Les objets sont les points du plan. Les configurations sont les triangles du plan. Un triangle est déterminé par ses sommets*

[1] En fait, puisque l'arrangement comporte des faces non bornées, le bon cadre est de plonger $\mathbb{R}^d$ de manière canonique dans l'espace projectif, puis de définir un ordre total sur l'espace projectif qui étende l'ordre lexicographique.

et en conflit avec tout point intérieur à son cercle circonscrit. Sous une hypothèse de position générale, les triangles de la triangulation de Delaunay d'un ensemble fini X de points sont les triangles actifs sur X.

Exercice 9.6 *Explicitez dans chacun des exemples ci-dessus la condition de position générale sur les objets de X afin que chaque configuration soit déterminée par un nombre borné d'objets, indépendant de la taille de X. Explicitez ces bornes.*

On note $\mathcal{P}(E)$ l'ensemble des parties d'un ensemble E.

Définition 9.7 *Un* espace de configurations *est un quadruplet $(\mathcal{O}, \mathcal{C}, \delta, \kappa)$ où*

1. *$\mathcal{O}$ est un ensemble (on dit aussi univers) dont les éléments sont appelés* objets,
2. *$\mathcal{C}$ est un ensemble, dont les éléments sont appelés* configurations,
3. *$\delta : \mathcal{C} \to \mathcal{P}(O)$ associe à chaque configuration un sous-ensemble d'objets. Une configuration c est dite* déterminée *par les objets de $\delta(c)$.*
4. *$\kappa : \mathcal{C} \to \mathcal{P}(O)$ associe à chaque configuration un sous-ensemble d'objets appelé son* ensemble de conflits. *Une configuration c est dite* en conflit *avec tout objet de $\kappa(c)$.*
5. *$\forall c \in \mathcal{C} : \delta(c) \cap \kappa(c) = \emptyset$.*

On appelle degré *d'un espace de configurations $(\mathcal{O}, \mathcal{C}, \delta, \kappa)$ le cardinal maximal de $\delta(c)$ pour toute configuration c.*

Soit X une partie de $\mathcal{O}$. L'espace de configurations $(\mathcal{O}, \mathcal{C}, \delta, \kappa)$ induit sur X un espace de configurations $(X, \mathcal{C}(X), \delta_X, \kappa_X)$ où
– *$\mathcal{C}(X)$ est l'ensemble des configurations de $\mathcal{C}$ déterminées par des parties de X (i.e. telles que $\delta(c) \subset X$),*
– *δ_X est la restriction de δ à $\mathcal{C}(X)$,*
– *et $\kappa_X : \mathcal{C}(X) \to \mathcal{P}(X)$ est définie par $\kappa_X(c) = \kappa(c) \cap X$.*
Une configuration $c \in \mathcal{C}$ est dite active *sur X si $c \in \mathcal{C}(X)$ et $\kappa(c) \cap X$ est vide. On note $\mathcal{C}^0(X)$ l'ensemble des configurations actives sur X.*

9.3 Algorithmes statiques

9.3.1 Formalisme et analyse randomisée

Un algorithme est dit *statique* ou *hors ligne* s'il s'applique à des données connues à l'avance. Dit autrement, les données sont chargées en mémoire une fois pour toute et ne seront plus modifiées. Relativement au formalisme des espaces de configurations, cela signifie qu'un ensemble fini X d'objets d'un univers $\mathcal{O}$ est donné, et que l'on cherche à construire la structure $\mathcal{S}(X)$ et donc en particulier les configurations actives de l'espace de configurations induit sur X. On notera $(X, \mathcal{C}(X), \delta, \kappa)$ cet espace de configurations.

Un algorithme incrémental et randomisé fonctionne de la manière suivante :
(1) On construit une permutation aléatoire $(x_1, x_2 \ldots, x_n)$ des éléments de X. On pose $X_i = \{x_1, x_2, \ldots, x_i\}$.

(2) On construit récursivement $\mathcal{S}(X_{i+1})$ (et donc l'ensemble des configurations actives sur X_{i+1}) en ajoutant x_{i+1} à $\mathcal{S}(X_i)$. Pour cela on doit supprimer les configurations actives à l'étape i (de $\mathcal{C}^0(X_i)$) en conflit avec x_{i+1} et ajouter les configurations *activées* par x_{i+1}, c'est-à-dire les configurations $c \in \mathcal{C}^0(X_{i+1})$ telles que $x_{i+1} \in \delta(c)$.

Pour faciliter la construction récursive, on introduit le *graphe des conflits* $\mathcal{G}_i$ entre les configurations actives courantes et les objets non-traités, défini par

$$\mathcal{G}_i = \big\{(c,x) \,:\, c \in \mathcal{C}^0(X_i) \wedge x \in \kappa(c)\big\}$$

En particulier, $\mathcal{G}_i \subset \mathcal{C}^0(X_i) \times (X \setminus X_i)$ est un graphe bipartite. On appellera *liste d'adjacences* ou *liste de conflits* d'un noeud de $\mathcal{G}_i$, la liste des voisins de ce noeud dans $\mathcal{G}_i$. On note $\ell_i(c)$ (resp. $\ell_i(x)$) la liste de conflits de la configuration c (resp. de l'objet x) dans $\mathcal{G}_i$. En particulier, $\ell_i(c) = \kappa(c)$. Ces listes permettent de retrouver rapidement les configurations actives de la i-ème étape qui vont disparaître suite à l'ajout de x_{i+1} : il s'agit précisément des configurations de $\ell_i(x_{i+1})$.

Le caractère statique des données permet de maintenir, en plus de la structure $\mathcal{S}(X_i)$, tout ou partie de ce graphe des conflits au cours de l'algorithme. L'objectif est de pouvoir reconstituer rapidement la liste $\ell_i(x_{i+1})$. Les détails propres au maintien (d'une partie) du graphe des conflits et de la structure $\mathcal{S}(X_i)$ dépendent du problème spécifique à traiter (décomposition trapézoïdale, enveloppe convexe, etc. . .). On peut néanmoins analyser de manière générique un algorithme incrémental randomisé et statique en supposant que le coût des diverses mises à jour à l'étape i vérifie

la condition de mise à jour statique : *le coût total de l'étape i lors de l'ajout de x_i est proportionnel au nombre de différences entre $\mathcal{G}_{i-1}$ et $\mathcal{G}_i$.*

Ce nombre de différences inclue

1. le nombre a_i de configurations activées par x_i, i.e. de $\mathcal{C}^0(X_i) \setminus \mathcal{C}^0(X_{i-1})$,
2. le nombre a'_i de conflits activés avec ces configurations (ce sont les nouvelles arêtes de $\mathcal{G}_i$ qui n'étaient pas dans $\mathcal{G}_{i-1}$),
3. le nombre t_{i-1} de configurations tuées par x_i, i.e. de $\mathcal{C}^0(X_{i-1}) \setminus \mathcal{C}^0(X_i)$,
4. le nombre t'_{i-1} de conflits tués avec ces configurations (ce sont les arêtes de $\mathcal{G}_{i-1}$ qui ne sont plus dans $\mathcal{G}_i$).

Puisque toute configuration ou tout conflit est nécessairement activé avant d'être tué et ne peut être tué qu'une fois au plus, on a compte tenu de $t_0 = t'_0 = t_n = t'_n = 0$:

$$\forall j \in [1,n] \,:\qquad \sum_{i=1}^{j} t_i \le \sum_{i=1}^{j} a_i \text{ et } \sum_{i=1}^{j} t'_i \le \sum_{i=1}^{j} a'_i. \tag{9.1}$$

Il en résulte que sous la condition de mise à jour statique, la complexité d'un algorithme incrémental statique est proportionnelle au nombre total de configurations et conflits activés au cours de l'algorithme. Pour tenir compte de l'aspect randomisé, on analyse la complexité en moyenne, relativement à la distribution uniforme sur les permutations des objets de X.

Lemme 9.8 *Soit un espace de configurations $(X, \mathcal{C}(X), \delta, \kappa)$ de $n = |X|$ objets et de degré $d = \max_{c\in\mathcal{C}(X)} |\delta(c)|$. On note $e_i = E(|\mathcal{C}^0(X_i)|)$ l'espérance du nombre de configurations actives à l'étape i. L'espérance du nombre total de configurations activées au cours de l'algorithme est majorée par*

$$d\sum_{i=1}^{n} \frac{e_i}{i}$$

L'espérance du nombre total de conflits activés au cours de l'algorithme est majorée par

$$d^2\sum_{i=1}^{n} \frac{n-i}{i^2} e_i$$

Preuve de la majoration de l'espérance du nombre de configurations actives : On pose $A_i = \mathcal{C}^0(X_i) \setminus \mathcal{C}^0(X_{i-1})$ l'ensemble des configurations activées à l'étape i et $a_i = |A_i|$. Fixons un sous-ensemble $X_i \subset X$ de i objets. L'évènement $\{X_i \text{ fixé }\}$ désigne l'ensemble des permutations de X dont l'*ensemble* des i premiers éléments coïncide avec X_i. On a ainsi

$$E(a_i \mid X_i \text{ fixé}) = \sum_{c\in\mathcal{C}^0(X_i)} P\big(c \in A_i \mid X_i \text{ fixé}\big) = \sum_{c\in\mathcal{C}^0(X_i)} P\big(x_i \in \delta(c) \mid X_i \text{ fixé}\big)$$

La dernière égalité traduisant le fait qu'une configuration active à l'étape i ne l'était pas à l'étape $i-1$ si et seulement si $x_i \in \delta(c)$. Or $P\big(x_i \in \delta(c) \mid X_i \text{ fixé}\big) \leq d/i$ car chaque objet de X_i a la même probabilité $1/i$ de se retrouver en i-ème position et, par définition du degré, au plus d objets sont dans $\delta(c)$. D'où

$$E(a_i \mid X_i \text{ fixé}) \leq \frac{d}{i}|\mathcal{C}^0(X_i)|$$

et le lemme 1.17 permet de conclure. □

Preuve de la majoration de l'espérance du nombre de conflits activés : On garde les notations précédentes et on note a'_i le nombre de conflits activés à l'étape i. On a ainsi

$$a'_i = \sum_{c\in A_i} |\kappa(c)| = \sum_{j>i} a_i^j$$

où $a_i^j = |A_i \cap \{c \in \mathcal{C}(X) : x_j \in \kappa(c)\}|$ désigne le nombre de configurations activées à l'étape i et en conflit avec x_j. Puisque x_j prend toutes les valeurs de $X \setminus X_i$ de manière équiprobable, l'espérance $E(a_i^j)$ ne dépend pas de j (cf. exercice 9.9 ci-dessous), i.e. $E(a_i^j) = E(a_i^{i+1})$ pour $j > i$. D'où

$$E(a'_i) = (n-i)E(a_i^{i+1})$$

On note $T_i = \mathcal{C}^0(X_i) \setminus \mathcal{C}^0(X_{i+1})$ l'ensemble des configurations actives tuées à l'étape $i+1$. Comme $a_i^{i+1} = |A_i \cap T_i|$, on a

$$E(a_i^{i+1}) = \sum_{c\in\mathcal{C}(X)} P\big(c \in A_i \wedge c \in T_i\big) = \sum_{c\in\mathcal{C}(X)} P\big(c \in A_i \mid c \in T_i\big)P\big(c \in T_i\big)$$

Soit X_i un sous-ensemble de i objets de X et $x \in X \setminus X_i$. On note $B(X_i, x)$ l'ensemble des permutations de X dont l'ensemble des i premiers éléments est X_i et dont le $i+1$-ième élément est x. On a égalité entre les événements $\{c \in T_i\}$ et $\bigcup B(X_i, x)$, l'union portant sur les (X_i, x) tels que $c \in \mathcal{C}^0(X_i)$ et $x \in \kappa(c)$. Or $P\big(c \in A_i \mid B(X_i, x)\big) \leq d/i$ par un raisonnement analogue à la preuve précédente. D'où $P\big(c \in A_i \mid c \in T_i\big) \leq d/i$ (cf. exercice 1.20). On en déduit

$$E(a'_i) \leq d\frac{n-i}{i}E(t_i)$$

où l'on a posé $t_i = |T_i|$. Par conséquent $E(\sum_i a'_i) \leq dE(\sum_i \frac{n-i}{i} t_i)$. On vérifie aisément que la décroissance de la fonction $i \mapsto \frac{n-i}{i}$ et les inégalités (9.1) impliquent (cf. exercice 9.10 ci-dessous)

$$E(\sum_i a'_i) \leq dE(\sum_i \frac{n-i}{i} a_i)$$

La première majoration du lemme sur l'espérance du nombre a_i de configurations activées à l'étape i permet de conclure. □

Exercice 9.9 *Soient X un ensemble de n objets, $k \in \mathbb{N}$ et $f : X^{k+1} \to \mathbb{R}$. On fixe un k-uplet d'indices distincts $I = (i_1, \ldots, i_k) \in [1, n]^k$ et on définit $n-k$ variables aléatoires $\{f_j\}_{j \notin I}$ sur l'ensemble $\mathcal{S}_X$ des permutations de X (muni de la probabilité uniforme) par*

$$\forall j \in [1, n] \setminus I\,, \forall \sigma = (x_1, \ldots, x_n) \in \mathcal{S}_X \ : \qquad f_j(\sigma) = f(x_{i_1}, \ldots, x_{i_k}, x_j)$$

Montrer que $E(f_j)$ ne dépend pas de j.

Exercice 9.10 *Soient deux familles de n nombres $(a_i)_{1 \leq i \leq n}$ et $(t_i)_{1 \leq i \leq n}$ telles que*

$$\forall j \in [1, n], \qquad \sum_{i=1}^{j} t_i \leq \sum_{i=1}^{j} a_i$$

et soit $(\alpha_i)_{1 \leq i \leq n}$ une suite positive décroissante. Montrer que

$$\sum_{i=1}^{n} \alpha_i t_i \leq \sum_{i=1}^{n} \alpha_i a_i$$

Références

- L'état de l'art [Mul00] et le livre [Mul94] de Mulmuley.

- Les articles fondateurs [CS89, CMS93].

- Le livre de Boissonnat et Yvinec [BY95].

La notion de graphe d'historique (ou d'influence) permet d'étendre le formalisme des algorithmes randomisés statiques (ou hors-ligne) aux algorithmes semi-dynamiques (ou en ligne) ou dynamiques. La version semi-dynamique permet d'ajouter au cours du temps des objets non-spécifiés au début de l'algorithme, ce qui n'est pas compatible avec la notion de graphe des conflits. Un exemple est fourni par l'algorithme de décomposition trapézoïdale de la section 6.2. Dans cet algorithme le graphe d'historique est donné par la structure $\mathcal{R}$ de recherche. Un algorithme dynamique gère également la suppression d'objets au cours du temps. Les références ci-dessus développent ces aspects.

9.3.2 Applications

On montre ici comment appliquer le formalisme précédent à des exemples de la section 9.2.

Décomposition trapézoïdale
On reprend le cadre de l'exemple 9.2. Soit alors une famille $X = \{x_1, x_2, \ldots, x_n\}$ de n segments du plan telle que deux segments sont soit disjoints soit s'intersectent en une extrémité commune. On considère ici que $\mathcal{S}(X)$ est un graphe d'adjacence sur $\mathcal{C}^0(X)$ où deux configurations sont adjacentes si les trapèzes correspondants partagent une paroi verticale. On note que sous une hypothèse raisonnable de position générale, le degré de l'espace de configurations est majoré par 4. Compte tenu de la taille linéaire de la décomposition trapézoïdale (i.e. de chaque $\mathcal{S}(X_i)$), le lemme 9.8 montre que le nombre moyen de configurations activées au cours de l'algorithme est en $O(n)$ et que le nombre moyen de conflits activés est un $O(n \log n)$ (il suffit de remplacer e_i par $O(i)$ dans les formules du lemme). Afin d'en déduire la même complexité pour tout l'algorithme de décomposition il faut valider la condition de mise à jour statique. En plus de la décomposition trapézoïdale $\mathcal{S}(X_i)$, on maintient ici le graphe des conflits $\mathcal{G}_i$ dans sa totalité. Ce graphe est codé sous forme de listes d'adjacences (de conflits) augmentées de pointeurs bidirectionnels, de sorte que si le trapèze τ est en conflit avec le segment x, l'occurrence de x dans $\ell(\tau)$ pointe vers l'occurrence de τ dans $\ell(x)$ et vice versa. Voyons les opérations à effectuer pour mettre à jour $\mathcal{S}(X_{i-1})$ et $\mathcal{G}_{i-1}$ suite à l'ajout du segment x_i. Il faut
- Supprimer les trapèzes courants en conflit avec x_i, c.-à-d. traversés par x_i, et supprimer les listes de conflits de ces trapèzes. Comme noté dans la section 9.3.1, ces trapèzes sont précisément ceux de la liste de conflits $\ell_{i-1}(x_i)$.
- Ajouter les trapèzes activés par x_i et les nouvelles relations d'adjacences liés à ces trapèzes et créer les listes de conflits pour ces trapèzes.

Pour effectuer ces opérations, on procède en deux étapes. Dans une première étape on se limite à subdiviser (en au plus quatre morceaux) chaque trapèze courant traversé par x_i ou par les parois verticales aux extrémités de x_i, quitte à conserver des parois verticales inutiles (typiquement toute paroi verticale courante coupée par x_i donne lieu à deux parois dont une seule doit être conservée). On modifie $\mathcal{S}(X_{i-1})$ et $\mathcal{G}_{i-1}$ en accord avec ces subdivisions. Pour cela on remplace chaque trapèze τ qui doit être subdivisé par les (au plus quatre) trapèzes qui le partitionnent et on répartie la liste de conflits $\ell_{i-1}(\tau) = \kappa(\tau)$ entre ces trapèzes. Ceci prend un temps $O(|\kappa(\tau)|)$. De plus, pour chaque segment $x \in \kappa(\tau)$, on remplace τ dans $\ell_{i-1}(x)$ par la sous-liste (de longueur trois au plus) *ordonnée de gauche à droite* des nouveaux trapèzes qui partitionnent τ et qui sont traversés par x. Ce qui prend à nouveau un temps $O(|\kappa(\tau)|)$. La condition sur l'ordre des trapèzes lors du remplacement permet de supposer que les trapèzes de la liste de conflits de tout segment sont ordonnés de gauche à droite le long du segment. Notons que les relations d'adjacences entre et avec les nouveaux trapèzes obtenus s'obtiennent aisément en temps proportionnel à leur nombre.

Dans une seconde étape, on supprime les morceaux de parois verticales inutiles, c.-à-d. qui ne passent pas par une extrémité de segment. Ceci revient à fusionner certain des trapèzes introduits à la première étape. Si l'on doit par exemple fusionner les trapèzes $\tau_1, \tau_2, \ldots, \tau_k$ en un trapèze τ' il faut fusionner les listes de conflits $\ell(\tau_1), \ell(\tau_2), \ldots, \ell(\tau_k)$

en une liste $\ell(\tau')$ et remplacer chaque τ_j par τ' dans toutes les listes $\ell(x)$ pour $x \in \ell(\tau_j)$. La difficulté consiste à supprimer en temps linéaire les répétitions lors de la fusion des listes et lors des remplacements. On utilise pour cela la propriété d'ordonnancement de gauche à droite des listes de conflits des segments. Plus précisément, on crée une liste $\ell(\tau')$ vide au départ. Puis, on parcourt chaque $\ell(\tau_j)$. Pour $x \in \ell(\tau_j)$, on place x dans $\ell(\tau')$ et on remplace dans $\ell(x)$ la sous-liste ℓ' de $\tau_1, \tau_2, \ldots, \tau_k$ (contenant τ_j) par τ' en temps proportionnel à $|\ell'|$. On supprime de plus x des listes de conflits des trapèzes de la sous-liste ℓ'. Ces suppressions évitent les répétitions dans $\ell(\tau')$. Le temps total pris pour cette seconde étape est clairement proportionnel à la somme des tailles des listes de conflits des trapèzes créés à la première étape. Et cette somme est elle-même proportionnel à la taille des listes de conflits tuées par x_i. On valide ainsi la condition de mise à jour statique.

Enveloppe convexe en dimension $d = 2$ ou 3

On rappelle que pour le problème de l'enveloppe convexe les objets sont les points de $\mathbb{R}^d$ et les configurations sont les paires (S, h) où h est un demi-espace et S est un ensemble de points tels que $\partial h = conv(S)$. Une configuration (S, h) est déterminée par S et en conflit avec les points contenus dans l'intérieur de h.

Soit alors une famille $X \subset \mathbb{R}^d$ de n points en position générale, c.-à-d. telle que toute sous-famille de $d+1$ points est affinement indépendante. On a vu que les hyperplans supports des facettes de l'enveloppe convexe de X correspondent aux régions actives sur X. Plus précisément on considère ici que $\mathcal{S}(X)$ est un graphe d'adjacence sur $\mathcal{C}^0(X)$ où deux configurations sont adjacentes si les facettes correspondantes partagent une $(d-2)$-face. En plus de l'enveloppe convexe $\mathcal{S}(X_i)$ on se contente ici de maintenir un petit sous-graphe de $\mathcal{G}_i$. Plus précisément on maintient pour chaque sommet $x \in X \setminus X_i$, une facette $f_i(x)$ de $\mathcal{S}(X_i)$ en conflit avec x, de sorte que $f_i(x) \in \ell_i(x)$. On pose $f_i(x) = NULL$ si x est intérieur à $\mathcal{S}(X_i)$. On maintient également pour chaque facette active, c, une liste de conflits $\ell'_i(c) \subset \ell_i(c)$ restreinte aux sommets x tels que $f_i(x) = c$. On maintient de plus un pointeur bidirectionnel entre x et son occurrence dans $\ell'_i(f_i(x))$. Puisque l'ensemble des facettes tuées par x forme un sous-graphe connexe de $\mathcal{S}(X_i)$, on peut identifier, par un simple parcours à partir de la facette $f_i(x)$, toutes les facettes en conflits avec x en un temps proportionnel à leur nombre (le degré des sommets de ce graphe est borné par hypothèse de position générale).

Afin d'appliquer les résultats du lemme 9.8 il faut valider la condition de mise à jour statique. Voyons les opérations à effectuer suite à l'ajout du sommet x_i pour mettre à jour $\mathcal{S}(X_{i-1})$ et les listes de conflits restreintes.

- Il faut supprimer les facettes courantes en conflit avec x_i. Comme décrit ci-dessus ces facettes s'obtiennent en temps proportionnel à leur nombre à partir de $f_{i-1}(x_i)$.
- Il faut ajouter les nouvelles facettes incidentes au sommet x_i. Soit $\{e_j\}_{j\in J}$ l'ensemble des $(d-2)$-faces bordantes, c.-à-d. incidentes à la fois à une facette supprimée et à une facette conservée. Notons que les faces bordantes e_j peuvent être déterminées lors du parcours des faces à supprimer. Les facettes activées par x_i sont de la forme $Conv(\{x_i\} \cup e_j)$. Pour $d = 2$, $\{e_j\}_{j\in J}$ est réduit à deux sommets et il est aisé d'établir les nouvelles relations d'adjacences entre arêtes actives. Pour $d = 3$, les arêtes dans $\{e_j\}_{j\in J}$ forment un cycle et leurs adjacences induisent celles entre les triangles activés correspondants.

– Pour chaque facette c supprimée, il faut mettre à jour les conflits avec les sommets de $\ell'_i(c)$. Soit $x \in \ell'_i(c)$. On parcourt les facettes actives en conflits avec x à partir de $f_{i-1}(x) = c$ en temps proportionnel à leur nombre. Si l'une c' de ces facettes n'est pas tuée par x_i, alors on pose $f_i(x) = c'$. Sinon, on a $\ell_{i-1}(x) \subset \ell_{i-1}(x_i)$ et ou bien x est intérieur à $\mathcal{S}(X_i)$, et alors $f_i(x) = NULL$, ou bien x est en conflit avec l'une des faces activées par x_i. Soit $c_j = Conv(\{x_i\} \cup e_j)$ une telle face. La face de $\mathcal{S}(X_{i-1})$ incidente à e_j et tuée par x_i est nécessairement en conflit avec x. Pour identifier c_j, il suffit donc, lors du parcours de $\ell_{i-1}(x)$, de tester si la face parcourue est incidente à une $(d-2)$-faces bordante e_j et dans ce cas de vérifier si $Conv(\{x_i\} \cup e_j)$ est en conflit avec x. Si aucun test n'aboutie, c'est que x est intérieur à $\mathcal{S}(X_i)$ et $f_i(x) = NULL$. Clairement, la mise à jour de $f_i(x)$ prend un temps proportionnel au nombre de faces tuées par x_i et en conflit avec x. Comme x n'apparaît que dans une unique liste $\ell'_i(c)$, on en déduit que la mise à jour de toutes les listes (restreintes) de conflits prend un temps proportionnel au nombre de conflits tués par x_i.

Finalement, l'ensemble des opérations à effectuer à l'étape i vérifie bien la condition de mise à jour statique. L'hypothèse de position générale implique que le degré de l'espace de configurations est précisément d. La complexité moyenne de l'algorithme ci-dessus est donc donnée par le lemme 9.8, ou encore par

$$O(d\sum_{i=1}^{n}\frac{e_i}{i} + d^2\sum_{i=1}^{n}\frac{n-i}{i^2}e_i)$$

Or, en dimension 2 ou 3 la taille de l'enveloppe convexe de X_i est un $O(i)$ (pour la dimension 3, on pourra utiliser la formule d'Euler). On en déduit que la complexité moyenne est un $O(n \log n)$ en dimension 2 ou 3. Notons que cette complexité est optimale en dimension 2 (et donc 3) par réduction du problème du tri à l'enveloppe convexe : le tri de n nombres t_i s'obtient en temps linéaire à partir de l'enveloppe convexe des n points du plan (t_i, t_i^2).

Bibliographie

[AE99] Pankaj K. Agarwal and Jeff Erickson. Geometric range searching and its relatives. In B. Chazelle, J.E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 23 of *Contemporary Mathematics*, pages 1–56. American Mathematical Society, 1999.

[BY95] J-D. Boissonnat and M. Yvinec. *Géométrie algorithmique*. Ediscience international, Paris, 1995. Version anglaise : Algorithmic geometry. Cambridge University Press, 1998.

[CF90] B. Chazelle and J. Friedman. A deterministic view of random sampling and its use in computational geometry. *Combinatorica*, 10 :229–249, 1990.

[Cha90] Bernard Chazelle. Lower bounds for orthogonal range searching : I. the reporting case. *Journal of the ACM*, 37(2) :200–212, 1990.

[CMS93] K. L. Clarkson, K. Mehlhorn, and R. Seidel. Four results on randomized incremental constructions. *Comp. Geom. : Theory and Applications*, 3 :185–212, 1993. J ai une version papier.

[CS89] K. Clarkson and P. Shor. Applications of random sampling in computational geometry, ii. *Discrete & Computational Geometry*, 4 :387–421, 1989.

[dBCvKO08] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry : Algorithms and Applications*. Springer-Verlag, third edition (march 2008) edition, 2008.

[GO04] J. E. Goodman and J. O'Rourke, editors. *Handbook of Discrete and Computational Geometry*. CRC Press, second edition edition, 2004.

[HP00] Sariel Har-Peled. Constructing planar cuttings in theory and practice. *SIAM J. Comput.*, 29(6) :2016–2039, 2000.

[Kir83] David Kirkpartrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12(1) :28–35, 1983.

[Mat91] Jiří Matoušek. Cutting hyperplane arrangements. *Discrete & Computational Geometry*, 6 :385–406, 1991.

[Mat92] Jiří Matoušek. Efficient partition trees. *Discrete & Computational Geometry*, 8 :315–334, 1992.

[Mat94] Jiří Matoušek. Geometric range searching. *ACM Computing Surveys*, 26(4) :421–461, 1994.

[Mat02] Jiří Matoušek. *Lectures on Discrete Geometry*. Number 212 in GTM. Springer Verlag, 2002.

[MT01] B. Mohar and C. Thomassen. *Graphs on Surfaces*. John Hopkins University Press, 2001.

[Mul94] Ketan Mulmuley. *Computational Geometry. An Introduction Through Randomized Algorithms*. Prentice Hall, 1994.

[Mul00] Ketan Mulmuley. Randomized Algorithms in Computational Geometry. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, chapter Chapter 16, pages 703–724. North-Holland, 2000.

[PA95] János Pach and Pankaj Agarwal. *Combinatorial Geometry*. John Wiley, 1995.

[SU00] J.-R. Sack and J. Urrutia, editors. *Handbook of Computational Geometry*. North-Holland, 2000.