Cours 3 — 24 janvier

Enseignant : Frédéric Magniez *Rédacteur : Jean Rougé*

3.1 Test d'associativité

Soit S un ensemble fini à n éléments, muni d'une loi de composition interne $\circ$.

But : On veut déterminer si $\circ$ est associative. On mesure la complexité en comptant le nombre d'opérations effectuées (au sens de $\circ$).

Théorème 3.1. *Il existe un algorithme probabiliste de type one-sided error répondant à ce problème en $\mathcal{O}(n^2)$ opérations*[1].

Preuve: Soit p un nombre premier ; notons $\mathcal{S} = \mathbb{Z}_p[S]$ le $\mathbb{F}_p$-espace vectoriel des vecteurs $x = (x_i)_{i \in S}$ à coordonnées dans $\mathbb{F}_p$.

On note, pour tout $i \in S$, e_i le *i-ième* vecteur de la base canonique de $\mathcal{S}$, et on munit $\mathcal{S}$ d'une loi de composition interne $\widetilde{\circ}$ définie sur la base canonique par

$$\forall (i,j) \in S^2,\ e_i \widetilde{\circ} e_j = e_{i \circ j}$$

puis étendue par linéarité à tout l'espace : pour tous les vecteurs $\left(x = \sum_{i \in S} x_i e_i, y = \sum_{j \in S} y_j e_j\right) \in \mathcal{S}^2$,

$$x \widetilde{\circ} y = \sum_{(i,j) \in S^2} x_i y_j e_{i \circ j}$$

Remarquons qu'on peut ré-écrire cette égalité sous la forme

$$x \widetilde{\circ} y = \sum_{k \in S} \left(\sum_{(i,j) \in S^2 / i \circ j = k} x_i y_j \right) e_k \tag{3.1}$$

On va montrer, en utilisant le lemme 3.2 ci-dessous que l'algorithme suivant répond aux spécifications du théorème :

1. C'est clairement mieux qu'un algorithme exhaustif naïf en n^3 opérations. On peut même se convaincre qu'on ne peut pas faire mieux ; on est bien obligés de construire la "table" de $\circ$: en effet, soient i_0 et j_0 deux éléments arbitraires de S, et soit $k_0 \neq i_0$, considérer par exemple la loi $\circ$ définie par

$$\begin{cases} i_0 \circ j_0 = k_0 \\ \forall (i,j) \neq (i_0, j_0),\ i \circ j = i_0 \end{cases}$$

ALGORITHME

1. Tirer au hasard trois vecteurs $(x, y, z) \in \mathcal{S}^3$
2. Calculer $[(x \,\widetilde{\circ} y) \,\widetilde{\circ} z - x \,\widetilde{\circ}\, (y \widetilde{\circ} z)]$
3. Renvoyer VRAI si ce vecteur est nul, FAUX sinon.

* Tout d'abord, sa complexité est bien en $\mathcal{O}(n^2)$ opérations sur S, d'après la définition de $\widetilde{\circ}$.

* Ensuite, d'après le lemme 3.2, si $\circ$ est associative, il renvoie toujours VRAI

* Supposons enfin que $\circ$ n'est pas associative, et calculons la probabilité d'erreur de notre algotithme.
Soient $\left(x = \sum_{i\in S} x_i e_i, y = \sum_{j\in S} y_j e_j, z = \sum_{k\in S} z_k e_k\right) \in \mathcal{S}^3$. Calculons, pour $l \in S$ fixé, la *l-ième* coordonnée du vecteur $(x \,\widetilde{\circ} y) \,\widetilde{\circ} z - x \,\widetilde{\circ}\, (y \widetilde{\circ} z)$, qu'on notera $P_l(x, y, z)$.
En utilisant (3.1), on obtient

$$P_l(x, y, z) = \left(\sum_{(i,j,k)\in S^3/i\circ(j\circ k)=l} x_i y_j z_k\right) - \left(\sum_{(i,j,k)\in S^3/(i\circ j)\circ k=l} x_i y_j z_k\right)$$

qui est donc un polynôme de degré 3 sur $\mathbb{F}_p$ (en $(x_i, y_j, z_k)_{(i,j,k)\in S^3}$). D'après le théorème de Schwartz-Zippel,

$$[P_l \neq 0] \Longrightarrow \left[\mathbb{P}_{(z,y,z)\in\mathcal{S}^3}(P_l(x, y, z) \neq 0) \geq 1 - \frac{\deg(P_l)}{|\mathbb{F}_p|} \geq 1 - \frac{3}{p}\right]$$

Or, comme $\circ$ n'est pas associative, d'après le lemme 3.2, il existe l dans S tel que $P_l \neq 0$. Ainsi, la probabilité d'erreur pour notre algorithme est inférieure à $\frac{3}{p}$.

□

Démontrons maintenant le lemme utilisé dans la preuve du théorème.

Lemme 3.2. *$\circ$ est associative dans S si et seulement si $\widetilde{\circ}$ est associative dans $\mathcal{S}$*

Preuve:

$\Rightarrow$ Calculons pour $\left(x = \sum_{i\in S} x_i e_i, y = \sum_{j\in S} y_j e_j, z = \sum_{k\in S} z_k e_k\right) \in \mathcal{S}^3$,

$$\begin{aligned}(x \,\widetilde{\circ} y) \,\widetilde{\circ} z &= \left(\sum_{(i,j)\in S^2} x_i y_j e_{i\circ j}\right) \widetilde{\circ} \left(\sum_{k\in S} z_k e_k\right) \\ &= \sum_{(i,j,k)\in S^3} x_i y_j z_k \underbrace{e_{i\circ j} \widetilde{\circ} e_k}_{=e_{(i\circ j)\circ k}}\end{aligned}$$

On obtient de même

$$x \,\widetilde{\circ}\, (y \widetilde{\circ} z) = \sum_{(i,j,k)\in S^3} x_i y_j z_k e_{i\circ(j\circ k)}$$

Or, comme $\circ$ est associative dans S, on a

$$\forall\,(i,j,k)\in S^3,\ (i\circ j)\circ k = i\circ(j\circ k)$$

d'où l'égalité voulue

$$(x\,\widetilde{\circ}\,y)\,\widetilde{\circ}\,z = x\,\widetilde{\circ}\,(y\widetilde{\circ}z)$$

$\Leftarrow$ Il suffit de remarquer que $S\subset\mathcal{S}$ par

$$i\rightsquigarrow e_i$$

□

3.2 Retour sur *k*-SAT

On rappelle brièvement le problème k-SAT introduit au cours 2 : on considère une formule $\varphi=\wedge_i C_i$ où les C_i sont des k-clauses, ie des unions de k littéraux choisis parmi n variables logiques $(x_i)_{1\leq i\leq n}$et leurs négations. Le but est de trouver une affectation a de ces variables logiques telles que $\varphi(a)=1$.

3.2.1 Walk-SAT itéré pour *3*-SAT

Considérons l'algorithme suivant :
ALGORITHME

1. Tirer au hasard (uniformément) une affectation $a\in\{0;1\}^n$
2. Répéter au plus $3n$ fois la boucle suivante :

   Si $\varphi(a)=1$, renvoyer a, et arrêter l'algorithme

   Sinon, choisir une clause C telle que $C(a)=0$, choisir un indice i tel que x_i apparaît dans C, et changer la valeur correspondante dans a (ie $a_i\leftarrow 1-a_i$)
3. Retour en 1.

Pour éviter les confusions, on parlera dans la suite d'*itérations* pour désigner le retour en 1 et le tirage aléatoire d'une nouvelle affectation initiale, par opposition aux *tours de boucle* internes à l'étape 2, avec une affectation initiale a donnée.

Théorème 3.3. *Il existe une constante K telle que si φ est satisfiable, pour toute affectation initiale a*

$$\underbrace{\mathbb{P}\,(\text{l'algorithme termine en moins de } 3n \text{ tours de boucle})}_{:=p} \geq \underbrace{\frac{K}{\sqrt{n}}\left(\frac{3}{4}\right)^n}_{:=q}$$

Preuve: Soit a une affectation initiale donnée ; posons $s \in \{0;1\}^n$ une solution de φ, et $X =$ *le nombre de bits différents entre a et s*. Notons par ailleurs

$$\forall j \in [\![0;n]\!],\ p_j = \mathbb{P}\left(\textit{l'algorithme termine en moins de 3n tours de boucle} \,|\, X = j\right)$$

On a par la formule de Bayes

$$p = \sum_{j=0}^{n} p_j \underbrace{\mathbb{P}\left(X = j\right)}_{=\frac{\binom{n}{j}}{2^n}}$$

Or, on a l'inégalité

$$p_j \geq \mathbb{P}\left(\textit{l'algorithme termine en exactement 3j tours de boucle} \,|\, X = j\right)$$

ce qui permet, en se ramenant à une marche aléatoire sur $\mathbb{Z}$ entier [2] d'écrire l'inégalité

$$\begin{array}{rcl} p_j & \geq & \mathbb{P}\left(\textit{l'algorithme effectue 2j pas à gauche et j pas à droite}\right) \\ & \geq & \binom{3j}{j}\left(\frac{2}{3}\right)^j\left(\frac{1}{3}\right)^{2j} \end{array}$$

la dernière inégalité venant du fait qu'on sélectionne une variable parmi les 3 de la clause non vérifiée, et qu'on a donc au moins une chance sur 3 que cette variable soit fausse (et même plus, puisque 2, voire les 3 variables peuvent être fausses).
Or, d'après la formule de Stirling,

$$\exists K \,/\, \binom{3j}{j} \sim \frac{K}{\sqrt{j}}\left(\frac{27}{4}\right)^j$$

d'où il vient pour n assez grand

$$\begin{array}{rcl} p & \geq & \sum_{j=0}^{n} \frac{\binom{n}{j}}{2^n}\binom{3j}{j}\frac{2^j}{3^{3j}} \\ & \geq & \sum_{j=0}^{n} \frac{K}{2^n}\binom{n}{j}\frac{1}{\sqrt{j}2^j} \\ & \geq & \frac{K}{\sqrt{n}}\left(\frac{3}{4}\right)^n \end{array}$$

□

Corollaire 3.4. *Dans l'algorithme précédent, l'espérance du nombre d'itérations est plus petite que*

$$\frac{1}{q} = \frac{\sqrt{N}}{k}\left(\frac{4}{3}\right)^n$$

On re-tombe sur des performances exponentielles, ce qui est plutôt normal, *3*-SAT étant NP-complet...

2. Cas clairement moins favorable que la marche aléatoire absorbante en 0 et repoussante en n que l'on considérait jusqu'à présent.

3.2.2 Généralisation à k-SAT

Théorème 3.5. *Avec le même algorithme, si φ est satisfiable, le nombre moyen d'itérations est majoré par $\left(\frac{1}{2}\left(1+\frac{1}{k}\right)\right)^{-n}$.*

Preuve: Reprenons les mêmes notations et les mêmes idées : on se ramène à une marche aléatoire sur $\mathbb{Z}$ tout entier, et on majore cette fois p_j par

$$\begin{array}{rcl} p_j & \geq & \sum_{i=0}^{j} \mathbb{P}\left(\textit{l'algorithme effectue } j+i \textit{ pas à gauche et } i \textit{ pas à droite}\right) \\ & \geq & \sum_{i=0}^{j} \binom{2i+j}{i} \left(\frac{k-1}{k}\right)^{i} \left(\frac{1}{k}\right)^{i+j} \\ & \geq & \left(1+\frac{1}{k}\right)^{j} \end{array}$$

d'où le résultat en sommant par la formule du binôme. □

3.2.3 Exercice : application à la 3-Coloriabilité

Soit $G=(X,E)$ un graphe non-orienté *3-coloriable*, c'est-à-dire qu'il existe une application $C : X \to [\![0;2]\!]$ telle que

$$\forall\, (x,y) \in X^2,\ (x,y) \in E \Rightarrow c(x) \neq c(y)$$

On cherche un algorithme qui colorie G avec seulement 2 couleurs, de telle sorte qu'aucun triangle ne soit mono-chromatique.

Soit une application $a : X \longrightarrow [\![0;1]\!]$; considérons l'ensemble T des triangles de G, c'est-à-dire les sommets $(i,j,k) \in X^3$ tels que

$$\begin{array}{rcl} (i,j) & \in & E \\ (j,k) & \in & E \\ (i,k) & \in & E \end{array}$$

et pour chaque triangle posons la clause logique $C_{i,j,k} = (a(i) \vee a(j) \vee a(k)) \wedge \left(\overline{a(i)} \vee \overline{a(j)} \vee \overline{a(k)}\right)$.

La formule logique

$$\varphi = \wedge_{(i,j,k)\in T} C_{i,j,k}$$

est une formule *3*-SAT satisfiable, puisqu'il suffit par exemple de poser

$$a : \left\{ \begin{array}{rcl} X & \longrightarrow & [\![0;1]\!] \\ x & \longmapsto & \left\{ \begin{array}{ll} 0 & \textit{si } C(x)=0 \\ 1 & \textit{sinon} \end{array} \right. \end{array} \right.$$

où C est le 3-coloriage existant par hypothèse ; on est donc ramenés à la résolution de *3*-SAT, qu'on peut par exemple traiter avec Walk-SAT.

3.3 S-T CONNECTIVITÉ

3.3.1 Cas général

Soient $G = (X, E)$ un graphe non-orienté et $(s, t) \in X^2$ deux sommets fixés de G. On cherche à déterminer s'il existe un chemin de s à t. On note $|X| = n$ et $|E| = m$

Il existe des algorithmes déterministes (programmation dynamique notamment) de complexité $\mathcal{O}(m+n)$ en temps, mais $\mathcal{O}(n)$ en espace, ce qui peut poser problèmé si le graphe est trop gros (graphe d'Internet par exemple). On cherche ici un algorithme plus lent, mais moins gourmand en mémoire.

Théorème 3.6. *Il existe un algorithme probabiliste de type one-sided error répondant à ce problème en $\mathcal{O}(mn)$ en temps et $\mathcal{O}(1)$ en espace.*

Preuve: Considérons l'algorithme de marche aléatoire suivant :
ALGORITHME

1. $u \leftarrow s$
2. Itérer tant que $u \neq t$

   Soit v voisin aléatoire de u (choisi uniformément)

   $u \leftarrow v$
3. Déclarer G S-T CONNEXE

* La complexité en espace est évidente.

* Si G n'est pas S-T CONNEXE, alors l'algorithme ne termine clairement pas.

* Reste à montrer la complexité en temps dans le cas où G est S-T CONNEXE.

Notons, pour tous sommets (i, j) dans la composante connexe de s, $h_{i,j}$ le nombre moyen d'étapes de la marche aléatoire pour aller de i à j ; et $C(G)$ le maximum pour $v \in X$ du temps moyen pour passer par tous les sommets de G en partant de v.

Remarquons qu'il existe un chemin $\tau = u_1 u_2 ... u_p$ de s à t de longueur majorée par $2n$: il suffit en effet de construire un arbre couvrant de la composante connexe de s et de le parcourir en profondeur.

On note un chemin $\sigma \geq \tau$ si τ est une sous-séquence de σ ; $C(G)$ est clairement majoré par le nombre d'étapes qu'il faut pour construire un chemin $\sigma \geq \tau$, donc par

$$C(G) \leq h_{u_1,u_2} + h_{u_2,u_3} + ... + h_{u_{p-1},u_p}$$

Comme la somme ci-dessus comporte au plus $2n$ termes, et que chacun est majoré par $2m$ d'après le lemme 3.7 ci-dessous, on obtient bien une complexité en $4nm = \mathcal{O}(mn)$.

□

Lemme 3.7. *Avec les notations de la preuve du théorème,*

i .

$$\forall i \in X,\ h_{i,i} = \frac{2m}{d(i)}$$

où $d(i)$ est le degré de i dans G, ie son nombre de voisins immédiats

ii . $\forall (i,j) \in X^2$,

$$(i,j) \in E^2 \Rightarrow h_{i,j} \leq 2m-1$$

Preuve: i . Le premier point est plus ou moins admis, mais on peut s'en convaincre par les arguments suivants : on formalise la marche aléatoire de l'algorithme par une chaîne de Markov de matrice de transition $P = (P_{i,j})_{(i,j)\in X^2} \in \mathcal{M}_n(\mathbb{R})$ où

$$\forall (i,j) \in X^2,\ P_{i,j} = \begin{cases} \frac{1}{d(i)} & si\ (i,j) \in E \\ 0 & sinon \end{cases}$$

Notons par ailleurs $X^k = \left(X_i^k\right)_{i\in X} \in \mathbb{R}^n$ le vecteur de probabilité d'état après la *k-ième* étape de marche aléatoire (ie pour tout sommet i, la probabilité de se trouver en i après k étapes vaut X_i^k), on a par définition

$$X^{k+1} = X^k P$$

On admet que si G est connexe (et sinon on peut simplement considérer la composante connexe de s), il existe une unique distribution stationnaire Π vérifiant

$$\Pi P = P$$

et on vérifie facilement par le calcul que la distribution Π est telle que

$$\forall i \in X,\ \Pi_i = \frac{d(i)}{2m}$$

Comme $X^k \underset{k\to+\infty}{\longrightarrow} \Pi$ (là encore, admis...), le lemme 3.8 nous convainc que pour tout sommet i, $h_{i,i} = \frac{1}{\Pi_i} = \frac{2m}{d(i)}$

ii . Soient deux sommets i et j tels que $(i,j) \in E$, remarquons que

$$\begin{aligned} h_{j,j} &= \sum_{k/(k,j)\in E} \mathbb{E}\left(\textit{nombre d'étapes pour aller de } k \textit{ à } j \mid \textit{la première étape en partant de } j \textit{ va en } k\right) \\ &= \frac{1}{d(j)} \sum_{k/(k,j)\in E} (1 + h_{k,j}) \end{aligned}$$

En particulier,

$$\frac{2m}{d(j)} = h_{j,j} \geq \frac{1}{d(j)} (1 + h_{i,j})$$

d'où

$$h_{i,j} \leq 2m-1$$

□

3.3.2 Exercices : exemples

1. Si G est complet, le résultat ci-dessus donne

$$C(G) \leq 2n^3$$

mais on peut obtenir beaucoup mieux : partons d'un sommet quelconque, et pour tout $i \in [\![1; n]\!]$, notons X_i la première étape où i sommets différents ont été visités. On cherche à calculer $C(G) = \mathbb{E}(X_n)$. Remarquons que $T_k := X_{k+1} - X_k$ suit une loi géométrique de paramètre $\frac{k}{n}$, et vérifie donc

$$\mathbb{E}(T_k) = \frac{n}{n-k}$$

Sommons

$$\begin{array}{rcl} \mathbb{E}(X_n) & = & \mathbb{E}(X_1) + \sum_{k=1}^{n-1} \mathbb{E}(T_k) \\ & = & N\left(\sum_{k=1}^{n} \frac{1}{k}\right) \end{array}$$

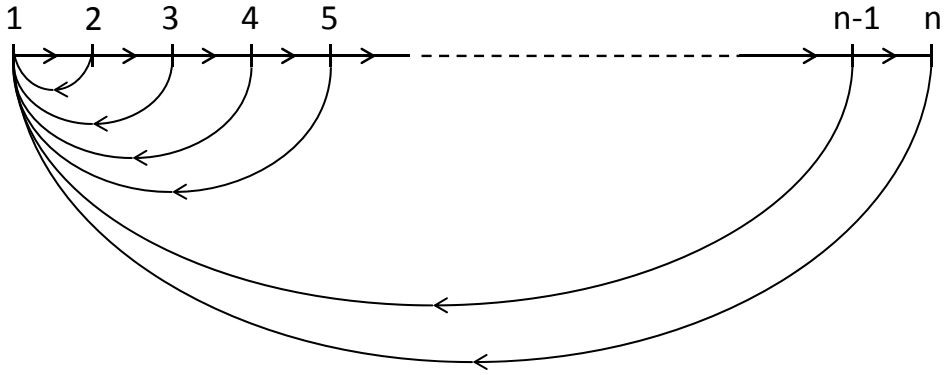
On retrouve la série harmonique, d'où

$$C(G) \sim n \ln n$$

2. Montrer qu'il existe un graphe orienté G tel que $C(G) \geq 2^n$.
Considérons le graphe $G = (X, V)$ défini par

$$\begin{cases} V = [\![1; n]\!] \\ \forall i \in [\![1; n-1]\!],\ (i, i+1) \in V \\ \forall j \in [\![1; n-1]\!],\ (j, 1) \in V \end{cases}$$

soit



En partant de 1, on voit que $C(G) = 2^n$

3. *Jeu du chat et de la souris :*
Soient un chat et une souris qui se déplacent de manière synchrone sur un graphe G. Montrer que le nombre moyen d'étapes avant qu'ils ne se rencontrent est majoré par $4nm^2$.

Considérons le graphe produit $G' = G \times G$, qui a n^2 sommets et m^2 arêtes ; on cherche à savoir quand on passera sur un sommet du type (x, x). Il y a donc n sommets "gagnants", et d'après le théorème précédent, le nombre moyen d'étapes est majoré par

$$\frac{1}{n} 4n^2 m^2 = 4nm^2$$

Retour sur un résultat courant

Lemme 3.8. *Soit $(X_i)_{i \in \mathbb{N}^*}$ une suite de variables aléatoires indépendantes identiquement distribuées selon la loi $\mathbb{P}(X_i = 1) = p = 1 - \mathbb{P}(X_i = 0)$.*

Notons $T = \min(i \in \mathbb{N}^* / X_i = 1)$. On a

$$\mathbb{E}(T) = \frac{1}{p}$$

Preuve: Calculons

$$\forall k \in \mathbb{N}^*, \mathbb{P}(T = k) = (1-p)^{k-1} p$$

d'où

$$\begin{array}{rcl} \mathbb{E}(T) & = & \sum_{k \in \mathbb{N}^*} k \mathbb{P}(T = k) \\ & = & p \sum_{k \in \mathbb{N}^*} k (1-p)^{k-1} \\ & = & \frac{p}{(1-(1-p))^2} \\ & = & \frac{1}{p} \end{array}$$

□