Introduction à l'algorithmique **Automne 2017**

IFT2125-6001 TA: Maëlle Zimmermann

Démonstration 12

1

Question: Calculer le nombre espéré de lancés de deux dés avant d'obtenir une paire de 1. Idem pour obtenir deux chiffres qui somment à 7.

Solution: Ce problème revient à calculer l'espérance d'une loi géométrique à paramètre p. En effet la loi géométrique considère le problème de répéter une épreuve dont la probabilité de succès est p, et la variable aléatoire géométrique dénote le rang du premier succès.

Soit p la probabilité d'obtenir l'évènement recherché lors d'un lancer. Le nombre espéré de lancers n est donné par:

$$\begin{aligned} n &= 1 \cdot p + 2(1-p)p + 3(1-p)^2 p + ... \\ &= \sum_{k=1}^{\infty} k(1-p)^{k-1} p. \end{aligned}$$

Or nous avons pour $x \in [0, 1)$:

$$f(x) = \sum_{k=0}^{\infty} x^k = \frac{1}{1-x} \qquad \text{(série géométrique)}$$

$$f'(x) = \sum_{k=1}^{\infty} k x^{k-1} = \frac{1}{(1-x)^2}.$$

Comme $n = p \sum_{k=1}^{\infty} k(1-p)^{k-1}$, nous obtenons

$$n = \frac{p}{(1-(1-p))^2} = \frac{1}{p}.$$

Comme la probabilité d'obtenir une paire de 1 est donnée par $p = 1/36$, le nombre espéré de lancers est 36. Similairement, la probabilité d'obtenir une paire de chiffres qui somment à 7 est $1/6$, donc le nombre espéré de lancers est 6.

2

Question: Une pièce biaisée produit des lancers pile avec probabilité $p \neq \frac{1}{2}$ et face avec probabilité $q = 1 - p$. Chaque lancer de pièce est indépendant des précédents. La valeur de p n'est pas connue. Trouver un procédé pour obtenir à partir de la pièce une séquence de bits aléatoire non biaisée.

Solution: Pour générer un bit aléatoire (0 ou 1), il suffit de faire deux lancers successifs de la pièce biaisée. Si deux piles ou deux faces sont obtenus, on recommence. Sinon,

- si on obtient pile puis face, on génère le bit 0,
- si on obtient face puis pile, on génère le bit 1.

Ces deux évènements ont une probabilité égale de se réaliser, à savoir pq. La probabilité d'obtenir l'un ou l'autre sachant que les deux lancers de pièces ont produit un pile et un face est $\frac{1}{2}$. On obtient une séquence de bits aléatoire non biaisée en répétant ce procédé.

3

Question: Considérer un problème qui a plus que deux réponses possibles. Il est alors possibles que pour certains instances du problèmes, plusieurs réponses soient correctes. Soit MC3 l'algorithme qui exécute un algorithme non-biaisé MC trois fois et retourne la réponse majoritaire s'il y lieu, ou, qui en cas d'égalité, retourne aléatoirement et uniformément l'une des réponses obtenues. Montrer que dans ce cas, l'amplification peut diminuer la probabilité que l'algorithme retourne une réponse correcte. Plus précisément, montrer qu'il existe un algorithme Monte-Carlo MC 75%-correct tel que MC3 n'est pas 71%-correct.

Solution: Soient $A_1, A_2, ..., A_n$ des bonnes réponses et B une mauvaise réponse. Soit MC l'algorithme qui retourne A_i avec probabilité $\frac{3}{4n}$ pour tout $i = 2, ..., n$, et B avec probabilité $\frac{1}{4}$. Notons que MC retourne une bonne réponse avec probabilité $n\frac{3}{4n} = \frac{3}{4}$ et est ainsi 75%-correct.

La probabilité que MC3 retourne une bonne réponse est $p(n) = f(n) + g(n) + \frac{2}{3}h(n)$ où

- $f(n)$ est la probabilité que MC3 tire trois bonnes réponses
- $g(n)$ est la probabilité que MC3 tire une mauvaise réponse et deux fois la même bonne réponse

- $h(n)$ est la probabilité que MC3 tire une mauvaise réponse et deux bonnes réponses différentes.

Nous avons

$$f(n) = \underbrace{\frac{3}{4}}_{\text{bonne réponse}} \quad \underbrace{\frac{3}{4}}_{\text{bonne réponse}} \quad \underbrace{\frac{3}{4}}_{\text{bonne réponse}}$$

$$g(n) = 3\Big(\underbrace{\frac{1}{4}}_{\text{mauvaise réponse}} \cdot \underbrace{\frac{3}{4} \cdot \frac{3}{4n}}_{\text{2 mêmes bonnes réponses}} \Big)$$

$$h(n) = 3\Big(\underbrace{\frac{1}{4}}_{\text{mauvaise réponse}} \cdot \underbrace{\frac{3}{4} \cdot \Big(\frac{3}{4} - \frac{3}{4n}\Big)}_{\text{2 bonnes réponses différentes}} \Big)$$

Ainsi,

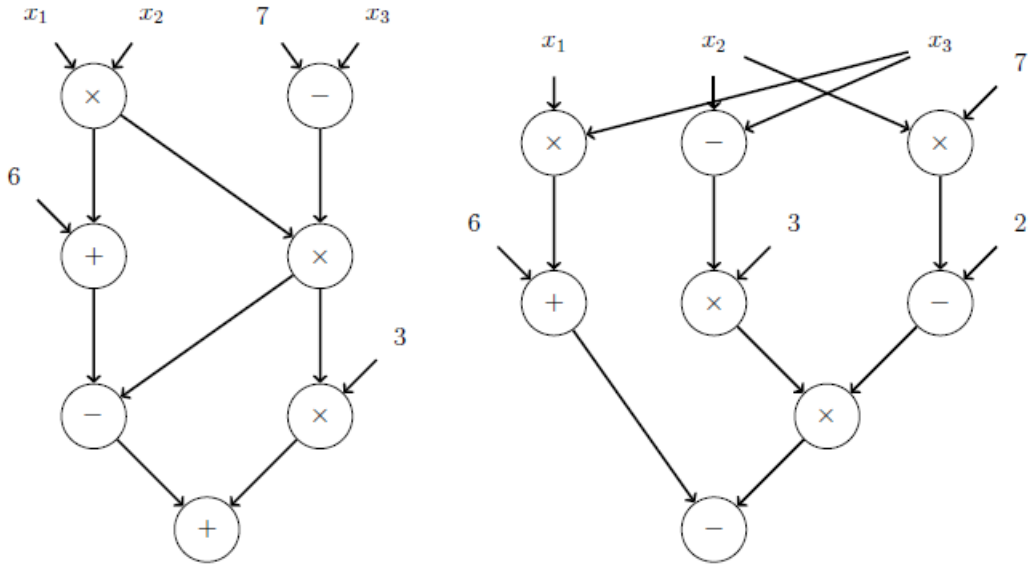
$$\begin{aligned} p(n) &= f(n) + g(n) + \frac{2}{3}h(n) \\ &= \Big(\frac{3}{4}\Big)^3 + \Big(\frac{3}{4}\Big)^2 \cdot \frac{3}{4n} + \frac{2}{3} \cdot \Big(\frac{3}{4}\Big)^2 \cdot \Big(\frac{3}{4} - \frac{3}{4n}\Big) \\ &= \frac{27}{64} + \frac{27}{64n} + \frac{3(3n-3)}{32n} \\ &= \frac{45n+9}{64n} \end{aligned}$$

Pour $n = 21$, nous avons $p(21) = 954/1344 = 159/224 < 0.7099$. Ainsi pour un problème avec 21 bonnes réponses, MC3 n'est pas 71%-correct. En fait, on peut même prouver que à partir de plus de 21 bonnes réponses possibles, l'algorithme MC3 n'est pas 71%-correct.

4

Question: Donner un algorithme de Monte Carlo qui permet de déterminer si deux polynômes p et q sur $\mathbb{Z}$, présentés par des circuits arithmétiques, sont égaux. Nous pouvons utiliser sans preuve le fait que le degré d'un polynôme est borné par la hauteur du circuit.

Solution: Considérons l'exemple suivant:



Les circuits C_p et C_q calculent respectivement les polynômes suivants:

$$p(x_1, x_2, x_3) = ((6 + (x_1x_2)) - (x_1x_2)(7 - x_3)) + 3(x_1x_2)(7 - x_3)$$
$$q(x_1, x_2, x_3) = (6 + (x_1x_3)) - 3(x_2 - x_3)(7x_2 - 2)$$

Afin de déterminer si $p = q$, nous posons $r = p - q$ et nous testons si r est nul. Autrement dit, nous testons si le circuit $C_r = C_p - C_q$ calcule le polynôme 0. Pour ce faire, nous choisissons une affectation aléatoire de $x_1, x_2, ..., x_m$, c'est-à-dire un ensemble de nombres $a_1, a_2, ..., a_m \in \mathbb{Q}$ et nous évaluons $r(a_1, a_2, ..., a_m)$. Si $r(a_1, a_2, ..., a_m) \neq 0$, alors il est impossible que r soit le polynôme 0. En revanche si $r(a_1, a_2, ..., a_m) = 0$, il est tout de même possible que r ne soit pas le polynôme 0, si on a choisi une racine de r.

Dans le cas où $m = 1$, c'est à dire que le polynôme est univarié, il suffit de choisir a_1 dans un ensemble de taille $k \cdot \deg(r)$ pour avoir une probabilité d'erreur au plus $1/k$. En effet, r possède au plus $\deg(r)$ racines.

Dans le cas multivarié, le nombre de racines de r n'est plus limité par $\deg(r)$. En fait, r peut posséder une infinité de racines. En effet, le polynôme $p(x_1, x_2, x_3) = x_1$ possède une infinité de racines de la forme $(0, x_2, x_3)$. Nous verrons grâce à un lemme que la stratégie du cas univarié se généralise tout de même aux polynômes multivariés.

Soient h_p et h_q la hauteur de C_p et C_q respectivement. Nous donnons l'algorithme suivant:

1. Poser $S = \{1, 2, ..., 4 \cdot \max(2^{h_p}, 2^{h_q})\}$
2. Choisir aléatoirement et indépendamment $a_1, a_2, ..., a_m \in \text{uniforme}(S)$

3. Poser $r = p - q$

4. Si $r(a_1, a_2, ..., a_m) = 0$ alors retourner oui, sinon retourner non.

Lorsque l'algorithme répond non, nous avons la certitude que p et q ne sont pas égaux. Lorsque l'algorithme répond oui, il est possible que p et q ne soient pas égaux et que l'algorithme se trompe. Pour évaluer la probabilité que l'algorithme se trompe dans le cas multivarié, nous avons besoin du lemme suivant:

Lemme de Schwartz-Zippel: soit $p \in \mathbb{Q}[x_1, x_2, ..., x_m]$ un polynôme non nul. Soient $S \subset \mathbb{Q}$ un sous-ensemble fini et $a_1, a_2, ..., a_m$ choisis dans S de façon uniforme et indépendante. Alors,

$$\Pr[p(a_1, a_2, ..., a_m) = 0] \leq \frac{\deg(p)}{|S|}. \tag{1}$$

Preuve: Le lemme est démontré par induction sur le nombre de variable.

Cas de base (m=1): dans ce cas, p est univarié et possède au plus $\deg(p)$ racines, donc on obtient que

$$\Pr[p(a_1) = 0] \leq \frac{\deg(p)}{|S|}.$$

Etape d'induction: nous pouvons mettre chaque puissance de x_1 en évidence dans p et obtenir la réécriture suivante.

$$p(x_1, x_2, ...x_m) = \sum_{i=0}^{d} x_1^i p_i(x_2, ...x_m)$$

Puisque p est non nul, il existe au moins un i tel que p_i est non nul. Soit i le plus grand tel indice, alors par hypothèse d'induction:

$$\Pr[p_i(a_2, ...a_m) = 0] \leq \frac{\deg(p_i)}{|S|} \leq \frac{\deg(p) - i}{|S|} \tag{2}$$

Si $p_i(a_2, ...a_m) \neq 0$, alors $p(x_1, a_2, ..., a_m)$ est un polynôme à une variable (x_1) de degré i, car i est le plus grand degré de x_1 non nul dans p. Ainsi par hypothèse d'induction:

$$\Pr[p(a_1, a_2, ..., a_m) = 0 \mid p_i(a_2, ...a_m) \neq 0] \leq \frac{i}{S} \tag{3}$$

Si on appelle A l'évènement $p(a_1, a_2, ..., a_m) = 0$ et B l'évènement $p_i(a_2, ...a_m) = 0$, alors on a

$$\begin{aligned}
\Pr[A] &= \Pr[A|B]\Pr[B] + \Pr[A|B^c]\Pr[B^c] \\
&\leq \Pr[B] + \Pr[A|B^c] \\
&\leq \frac{\deg(p) - i}{|S|} + \frac{i}{|S|} \quad \text{par (2) et (3)} \\
&= \frac{\deg(p)}{|S|}.
\end{aligned}$$

Ce qui conclut la preuve. □

Considérons maintenant le cas où l'algorithme ci-haut réponde oui. Soit w la probabilité que l'algorithme se trompe, autrement dit que $r(a_1, a_2, ...a_m) = 0$ sachant que r est non nul. Nous avons:

$$\begin{aligned}
w &\leq \frac{\deg(r)}{|S|} && \text{par (1)} \\
&\leq \frac{\max(\deg(p), \deg(q))}{|S|} && \text{car } \deg(r) \leq \max(\deg(p), \deg(q)) \\
&= \frac{\max(\deg(p), \deg(q))}{4 \cdot \max(2^{h_p}, 2^{h_q})} && \text{par déf. de } S \\
&\leq \frac{\max(\deg(p), \deg(q))}{4 \cdot \max(\deg(p), \deg(q))} && \text{car } \deg(p) \leq 2^{h_p} \text{ et } \deg(q) \leq 2^{h_q} \\
&= \frac{1}{4}
\end{aligned}$$

Ainsi l'algorithme se trompe avec probabilité au plus $\frac{1}{4}$ lorsqu'il répond oui.

Analyse du temps d'exécution: Nous analysons le temps d'exécution de l'algorithme selon la taille de l'entrée, i.e. selon le degré de p et q et selon le nombre de variables m. Si on peut tirer à pile ou face en $O(1)$, il est possible de sélectionner un nombre a_i dans S par recherche binaire en temps $\log_2 |S| = \log_2(4 \cdot \max(2^{h_p}, 2^{h_q})) = 2 + \max(h_p, h_q)$ tirs à pile ou face. Le temps d'exécution des algorithme de sélection aléatoire est donc dans $O(m \cdot \max(h_p, h_q))$. Evaluer le circuit r se fait en temps polynômial. Le temps d'exécution de l'algorithme est donc polynomial dans la taille de l'entrée.