Sujets de révisions pas étoile : éléments de correction

Logique

Exercice 1

On introduit trois variables propositionnelles A, B, M dont la sémantique est "A est un Pur", "B est une Pur" et "nous sommes sur l'île Maya".

1. a) Les dires de A se traduisent en $B \wedge M$ et ceux de B en $\neg A \wedge M$. Les contraintes de l'énoncé imposent que la formule $F_1 = (A \Leftrightarrow (B \wedge M)) \wedge (B \Leftrightarrow (\neg A \wedge M))$ est vraie.

   b) On utilise les règles du calcul propositionnel (transformation d'une implication, règles de De Morgan, distributivité...) et on obtient que :

$$F_1 \equiv (\neg A \vee B) \wedge (\neg A \vee M) \wedge (\neg B \vee \neg M \vee A) \wedge (\neg B \vee \neg A) \wedge (\neg B \vee M) \wedge (A \vee \neg M \vee B)$$

   c) Dans l'arbre obtenu, une seule feuille est étiquetée par $\top$: celle correspondant à la valuation v telle que $v(A) = v(B) = v(M) = 0$. On en déduit que A et B sont deux Pires et que le voyageur n'est pas sur l'île Maya.

2. a) De manière similaire, on obtient $F_2 = (1) \wedge (2)$ avec $(1) = A \Leftrightarrow (\neg A \wedge \neg B \wedge M)$ représentant les contraintes dues aux dires de A et $(2) = B \Leftrightarrow (\neg A \vee \neg B) \wedge \neg M$ celles dues à B.

   b) On obtient la table de vérité suivante :

A	B	M	$\neg A \wedge \neg B \wedge M$	(1)	$(\neg A \vee \neg B) \wedge \neg M$	(2)	F_2
0	0	0	0	1	1	0	0
0	0	1	1	0	0	1	0
0	1	0	0	1	1	1	1
0	1	1	0	1	0	0	0
1	0	0	0	0	1	0	0
1	0	1	0	0	0	1	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	1	0

   On en déduit que F_2 est équivalente à la formule $F = \neg A \wedge B \wedge \neg M$ qui est bien une formule sous forme normale disjonctive (à une clause).

   c) D'après la question précédente, le voyageur n'est toujours pas arrivé. D'ailleurs, on connaît aussi la nature de A et B : A est Pire et B est Pur.

Exercice 2

1. On a :

Symbole	Nature	Arité
f	Relation	3
g	Fonction	2
a	Fonction	0 (contante)
Q	Relation	1

2. Les termes sont a, z, x, y, $g(x, y)$, $Q(a)$ et $g(x, Q(a))$. Les formules atomiques sont $f(g(x, y), a, z)$ et $f(x, g(x, Q(a)), z)$.

3. La première occurrence de x est liée par le premier $\forall$; les deux suivantes sont libres. La seule occurrence de y est liée par $\exists$. La première occurrence de z n'est pas liée et la seconde l'est par le second $\forall$.

Exercice 3

1. a) L'arbre de déduction suivant conclut :

$$\dfrac{\dfrac{}{\neg P \vee Q, P \vdash \neg P \vee Q}\ \text{ax} \qquad \dfrac{}{\neg P \vee Q, P, Q \vdash Q}\ \text{ax} \qquad \dfrac{\dfrac{\dfrac{}{\neg P \vee Q, P, \neg P \vdash P}\ \text{ax} \qquad \dfrac{}{\neg P \vee Q, P, \neg P \vdash \neg P}\ \text{ax}}{\neg P \vee Q, P, \neg P \vdash \bot}\ \text{elim-}\neg}{\neg P \vee Q, P, \neg P \vdash Q}\ \text{elim-}\bot}{\dfrac{\neg P \vee Q, P \vdash Q}{\vdash (\neg P \vee Q) \Rightarrow (P \Rightarrow Q)}\ \text{intro-}\Rightarrow\ (2\text{ fois})}\ \text{elim-}\vee$$

   Pour en avoir l'idée, on peut tenir le raisonnement "mathématique" suivant : on sait que $\neg P \vee Q$ et on veut montrer $P \Rightarrow Q$. Suppsons donc P et montrons Q. Comme $\neg P \vee Q$, soit on a Q et dans ce cas c'est gagné, soit on a $\neg P$ mézalors comme on a aussi P on en déduit $\bot$ et de là on peut déduire tout ce qu'on veut ; par exemple Q.

   b) La question est de savoir si le séquent $\vdash (P \Rightarrow Q) \Rightarrow (\neg P \vee Q)$ est dérivable. C'est le cas car ce séquent est sémantiquement vrai : comme la déduction naturelle est complète (tout ce qui est sémantiquement vrai est syntaxiquement prouvable) on en déduit qu'une preuve de ce séquent existe sans même avoir à la produire.

2. a) On note $F = \forall x A \vee \forall x B$. L'arbre de déduction suivant conclut :

$$\dfrac{\dfrac{}{F \vdash F}\ \text{ax} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{F, \forall x A \vdash \forall x A}\ \text{ax}}{F, \forall x A \vdash A}\ \text{elim-}\forall}{F, \forall x A \vdash A \vee B}\ \text{intro-}\vee}{f, \forall x A \vdash \forall x (A \vee B)}\ \text{intro-}\forall \qquad \dfrac{\text{même raisonnement}}{F, \forall x B \vdash \forall x (A \vee B)}}{\dfrac{F \vdash \forall x (A \vee B)}{\vdash F \Rightarrow \forall x (A \vee B)}\ \text{intro-}\Rightarrow}\ \text{elim-}\vee$$

   Dans la deuxième branche, l'utilisation de la règle elim-$\forall$ se fait en substituant x par x dans A et l'utilisation de la règle intro-$\forall$ est licite car x n'est ni libre ni dans F ni dans $\forall x A$.

3. La déduction naturelle un système de déduction correct (tout ce qu'on peut syntaxiquement y prouver est sémantiquement vrai). La formule $\forall x (A \vee B) \Rightarrow \forall x A \vee \forall x B$ n'est pas vraie donc elle n'est pas prouvable en déduction naturelle.

Exercice 4

1. Immédiat.

2. On a $N_0 = N_1 = 1$ et pour tout $n \geq 2$, $N_n = N_{n-1} + N_{n-2} + 1$ (on fait un appel qui déclenche un appel sur $n-1$ et un sur $n-2$). On montre la deuxième partie de la question par récurrence sur n.

3. L'ordre de grandeur de la complexité de l'algorithme récursif sur l'entrée n est N_n ; d'après la question précédente il suffit de déterminer explicitement f_n pour l'estimer. On étudie la suite récurrente d'ordre deux qu'est f_n et on trouve que $N_n = \Theta((1+\sqrt{5})/2)^n)$. Comme $(1+\sqrt{5})/2 > 1$, on en déduit que la complexité de l'algorithme récursif est exponentielle ce qui n'est guère satisfaisant.

4. Classique. On peut même obtenir un espace constant en prime.

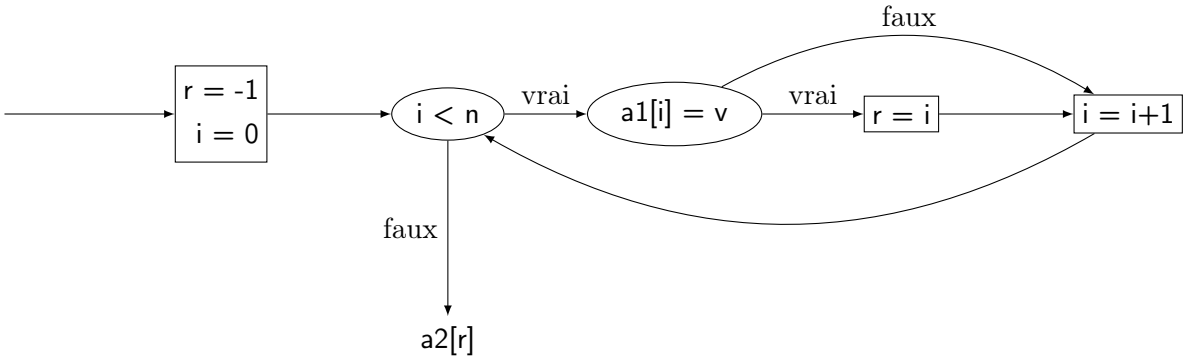
5. On constate et on montre par récurrence que pour tout $n \geq 2$ on a :

$$F^n = \begin{pmatrix} f_n & f_{n-1} \\ f_{n-1} & f_{n-2} \end{pmatrix}$$

6. Pour déterminer f_n, il suffit donc de calculer F^n et d'en récupérer le terme en haut à gauche. Pour ce faire, on n'aura besoin que de faire des produits matriciels entre des matrices de taille 2×2 et un tel produit se fait en temps constant. En exploitant la même stratégie que pour l'exponentiation rapide de nombre, on peut se contenter de faire $O(\log n)$ tels produits d'où le résultat.

Exercice 5

1. Lorsqu'il ne provoque pas d'accès en dehors du tableau `a2`, il renvoie la valeur située dans `a2` à l'indice contenant pour la dernière fois `v` dans `a1`.
2. On obtient :



3. Pas de difficulté.
4. Ce chemin couvre tous les arcs et tous les sommets.
5. A priori non car ce test échoue à exhiber le problème suivant : lorsque `v` n'apparaît pas dans `a1`, la valeur renvoyée est `a2[-1]` car `r` n'est jamais modifié ce qui est problématique.

Exercice 6

1. Pour chaque élément du tableau on compte son nombre d'occurrences en parcourant le tableau (ce qui nécessite $O(n)$ opérations) et on le compare à $n/2$. Au pire cas il n'y a pas d'élément majoritaire dans le tableau et pour s'en rendre compte on fait n parcours du tableau pour un coût total en $O(n^2)$.
2. Immédiat, complexité en $\Theta(j - i)$.
3. On note A_g et A_d les deux morceaux du tableau et x_g et x_d leur élément (il y a évidemment unicité d'un élément majoritaire par définition) majoritaire éventuel. On remarque que si A admet un élément majoritaire alors (A_g admet un élément majoritaire ou A_d admet un élément majoritaire). Ainsi :
   - Si ni A_g ni A_d n'a d'élément majoritaire, alors A n'en a pas non plus.
   - Si x_g existe mais pas x_d, on compte le nombre d'occurrence de x_g dans A : s'il y en a strictement plus de $n/2$, x_g est majoritaire dans A sinon A n'a pas d'élément majoritaire.
   - Le cas où x_d existe mais pas x_g est symétrique.
   - Si x_g et x_d existent, c'est l'élément majoritaire de A si $x_g = x_d$ et A n'a pas d'élément majoritaire sinon.

Attention : dans le cas où seul l'un des deux sous tableaux produit un élément majoritaire qui est donc candidat à être majoritaire dans A, il faut bien recompter son nombre d'occurrences dans A en entier pour pouvoir conclure.

4. C'est la traduction de la question précédente en pseudo-code.

5. Sur une entrée de taille n, majoritaire effectue deux appels à majoritaire sur un tableau de taille $n/2$ et dans le pire cas (correspondant aux cas 2 et 3 de la question 3) fait ensuite un appel à occurrences sur un tableau de taille n. Donc :
$$C(n) = 2C\left(\frac{n}{2}\right) + \Theta(n)$$
On en déduit une complexité en $O(n \log n)$ pour majoritaire.

Exercice 7

1. La somme maximale obtenable est $3 + 7 + 4 + 9 = 23$.

2. A chaque étage du triangle, on peut se déplacer sur 2 chemin : il y a donc 2^n chemins possibles. Calculer la somme d'un chemin peut se faire en $O(n)$ en le parcourant : on obtiendrait donc un algorithme naïf consistant à calculer naïvement la somme de tous les chemins en $O(n2^n)$.

3. Soit $i, j \in [\![0, n]\!]$. Pour $j > i$, le sommet (i, j) n'existe pas mais il est légitime d'imposer $f(i, j) = 0$ vu la suite. En effet, si $i \neq 0$ et $i \leq j$ on a
$$f(i, j) = T(i, j) + \max(f(i - 1, j), f(i - 1, j + 1))$$
où $T(i, j)$ désigne le coefficient en position (i, j) dans la matrice représentant le triangle. En effet, il n'y a que deux chemins permettant d'accéder à la position (i, j) : celui "à gauche" qui passe par $(i - 1, j)$ et celui "à droite" qui passe par $(i - 1, j + 1)$. Il ne reste plus qu'à déterminer la valeur de $f(0, 0)$ qui est évidemment égale à $T(0, 0)$.

4. L'algorithme dynamique demandé découle immédiatement de la relation de récurrence : on remplit une matrice de taille $n \times n$ avec la valeur de $f(i, j)$ en position (i, j) ligne par ligne par ordre croissant des i. Puis on calcule le max de la dernière ligne de cette matrice. On obtient alors un algorithme en $O(n^2)$ en temps comme en espace.

   En remarquant que le calcul du coefficient (i, j) de cette matrice ne nécessite de connaître que des coefficients sur sa $i - 1$-ème ligne, on peut même ramener la complexité spatiale à un $O(n)$.

5. Si on a choisi l'option où on stocke toute la matrice des $f(i, j)$, il est facile d'obtenir un chemin de somme maximal sans surcoût : il suffit de calculer le maximum de la dernière ligne de cette matrice puis de déterminer le plus grand des deux éléments adjacents à ce maximum puis le plus grand des deux éléments adjacents à ce dernier... jusqu'à aboutir au sommet. Ceci n'exige que $O(n)$ opérations et est donc absorbé par le $O(n^2)$ obtenu précédemment.

Exercice 8

1. Pour $n \geq 2$, notons I_n l'instance de SUBSET SUM suivante : $C = n$ et $T = [1, n]$. L'algorithme proposé renvoie sur cette instance le sous ensemble $\{1\}$ de somme $S_n = 1$ alors qu'une solution optimale consisterait en le sous ensemble $\{n\}$ de somme $S_n^\star = n$.

   Si cet algorithme était un algorithme d'approximation à facteur constant, il existerait $\alpha > 0$ tel que pour tout $n \geq 2$ on ait $\alpha S_n^\star \leq S_n$. Mais ce qui précède montre en passant à la limite que $\alpha \leq 0$!

2. Le tri peut se faire en $O(n \log n)$ qui est aussi la complexité globale car la boucle se fait en $O(n)$.

3. On note S la somme obtenue avec l'algo et $S^\star$ la somme optimale pour une instance donnée :

   - Si la somme de tous les t_i est inférieure à C alors $S = S^\star \geq S^\star/2$.

- Si aucun des t_i n'est inférieur à C alors $0 = S = S^\star \geq S^\star/2$.
- Sinon, notons S_{courant} la somme courante au moment où dans l'algorithme on a déjà ajouté un t_j à la varibale S puis rencontré un t_i qui ne peut pas y être ajouté sans dépasser C. Montrons que $S_{\text{courant}} \geq C/2$.

  Sinon, on aurait $S_{\text{courant}} < C/2$. Vu l'ordre dans lequel sont considérés les t_k, tous ceux qui interviennent dans S_{courant} sont supérieurs à t_i. Ainsi $C/2 > S_{\text{courant}} \geq t_k \geq t_i$ mais dans ces conditions on peut ajouter t_i à S_{courant} sans dépasser C ce qui est impossible.

  On en déduit que $S \geq S_{\text{courant}} \geq C/2 \geq S^\star/2$ car au mieux $S^\star = C$.

On a conclut dans tous les cas.

4. Non, 1/2 est le meilleur facteur d'approximation pour cet algorithme. Pour tout $n \geq 1$, notons I_n l'instance donnée par $C = 2n$ et $T = [n+1, n, n]$. La somme trouvée par l'algorithme glouton est $S_n = n + 1$; la somme optimale est $S_n^\star = 2n$.

   Si on A admet un facteur d'approximation α, on a en particulier pour tout $n \geq 1$,

$$\alpha \leq \frac{S_n}{S_n^\star} = \frac{n+1}{2n}$$

   et en passant à la limite on a $\alpha \leq 1/2$.

Exercice 9

1. La recherche dans un ABR se fait en $O(h)$ où h est la hauteur de l'arbre considéré donc oscille entre un $O(\log n)$ pour les arbres équilibrés et un $O(n)$ pour les arbres peignes par exemple.

   Remarque : La question est mal posée : la complexité pire cas est en $O(h)$ quel que soit l'ABR et la complexité meilleur cas en $O(1)$ puisque si on a de la chance l'élément cherché est à la racine !

   L'ordre d'insertion $1, 2, 3, 4, 5, 6, 7$ produit un arbre peigne et l'ordre $4, 2, 6, 1, 3, 5, 7$ un arbre complet.

2. Si $R_n = i$, par définition de R_n le sous arbre gauche de l'ABR considéré est un ABR aléatoire contenant $i - 1$ éléments et le sous arbre droit un ABR aléatoire contenant $n - i$ éléments. donc $H_n = 1 + \max(H_{i-1}, H_{n-i})$ puis $X_n = 2^{H_n} = 2\max(H_{i-1}, X_{n-i})$.

3. $Z_{n,i}$ vaut 1 pour exactement une valeur de $i \in [\![1, n]\!]$: la question précédente conclut.

4. On a :

$$\begin{aligned}
\mathbb{E}[X_n] &= 2\sum_{i=1}^{n} \mathbb{E}[Z_{n,i}\max(X_{i-1}, X_{n-i})] \quad \text{par linéarité et la question 3} \\
&= 2\sum_{i=1}^{n} \mathbb{E}[Z_{n,i}]\mathbb{E}[\max(X_{i-1}, X_{n-i})] \quad \text{par l'indépendance rappelée par l'énoncé} \\
&= 2\sum_{i=1}^{n} \frac{1}{n}\mathbb{E}[\max(X_{i-1}, X_{n-i})] \quad \text{car l'équiprobabilité des permutations donne } \mathbb{P}(R_n = i) = 1/n \\
&\leq 2\sum_{i=1}^{n} \frac{1}{n}(\mathbb{E}[X_{i-1}] + \mathbb{E}[X_{n-i}]) \quad \text{par positivité des } X_i \\
&= \frac{4}{n}\sum_{i=0}^{n-1} \mathbb{E}[X_i] \quad \text{en réindiçant}
\end{aligned}$$

5. D'après l'énoncé on a :

$$\mathbb{E}[X_n] \leq \frac{1}{4}\binom{n+3}{n} = \frac{1}{4}\frac{(n+3)(n+2)(n+1)}{2} \leq Kn^3$$

lorsque n est suffisamment grand (avec K un constante indépendante de n). Comme $x \mapsto 2^x$ est convexe, l'inégalité de Jensen affirme que :

$$2^{\mathbb{E}[H_n]} \leq \mathbb{E}[2^{H_n}] = \mathbb{E}[X_n] \leq Kn^3$$

et en prenant le $\log_2$ de cette inégalité on aboutit à : $\mathbb{E}[H_n] \leq \log_2(Kn^3) = O(\log n)$. Autrement dit, si on construit aléatoirement un ABR à n noeuds, en espérance il est équilibré.

Exercice 10

1. RAS.
2. On doit trouver : didondinaditondedodusdindons
3. On obtient le dictionnaire suivant :

Chaîne	A	B	C	AA	AB	BA	AAC	CB	BAA	AAA	ABB	BC	CBA	AC
Code	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Le résultat de la compression est : 1 1 2 4 3 6 4 5 2 8 1 3

4. On doit trouver : papouspapasapouxpapas

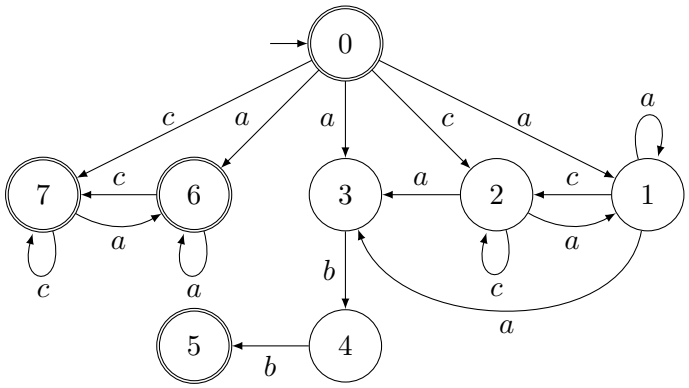
Exercice 11

RAS

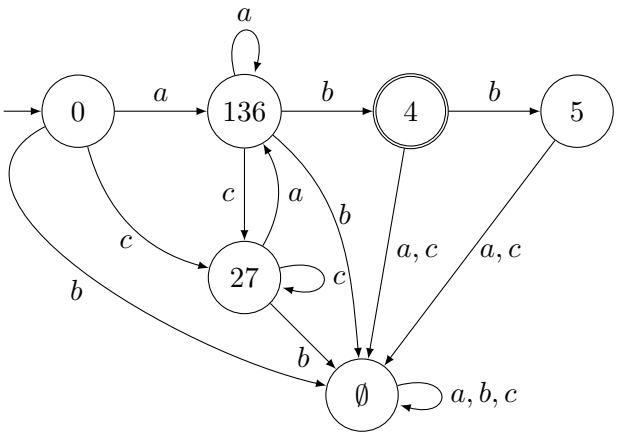
Exercice 12

1. a) On linéarise en $(a_1 + c_1)^\star a_2 b_1 b_2 + (a_3 + c_2)^\star$. On a :
   - $P(L) = \{a_1, c_1, a_2, a_3, c_2\}$.
   - $D(L) = \{b_2, a_3, c_2\}$.
   - $F(L) = \{a_1a_1, a_1c_1, c_1c_1, c_1a_1, a_1a_2, c_1a_2, a_2b_1, b_1b_2, a_3a_3, a_3c_2, c_2c_2, c_2a_3\}$.

   D'où l'automate suivant (dont on rend l'état initial final puisque $\varepsilon \in L$).



   b) Il suffit de déterminiser et compléter l'automate précédent puis d'échanger les états finaux et non finaux :



2. a) On devrait trouver $b^\star a^+ b\,((a + b^+ a)a^\star b)^\star$.

   b) C'est l'automate des occurrences pour ab donc il reconnaît le langage dénoté par $(a + b)^\star ab$.

Exercice 13

RAS

Exercice 14

1. (1), (2) et (4) sont vraies par double inclusion. (3) est faux : on a $A(B \cap C) \subset AB \cap AC$ mais pas la réciproque. Avec $A = \{a, ab\}$, $B = \{bb\}$ et $C = \{b\}$ par (contre)exemple, on a $A(B \cap C) = \emptyset$ mais $abb = a \cdot bb = ab \cdot b \in AB \cap AC$.

2. (1) est vrai : si $u \in L$ alors $uu \in L^2$ donc $u \in \sqrt{L^2}$ par définition.

   (2) est faux : avec $L = \{\varepsilon, aa\}$, on a $\varepsilon aa = aa \in L^2$ donc $a \in \sqrt{L^2}$ mais $a \notin L$.

   (3) est faux : avec $L = \{a\}$, on a $\sqrt{L} = \emptyset$ et $(\sqrt{L})^2 = \emptyset$ aussi.

   (4) est faux : avec $L = \{aa, bb\}$, on a $\sqrt{L} = \{a, b\}$ et $(\sqrt{L})^2 = \{aa, ab, ba, bb\} \not\subseteq L$;

   (5) est faux : avec $L = \{aa, bb\}$ on a $L^2 = \{aaaa, aabb, bbaa, bbbb\}$ donc $\sqrt{L^2} = \{aa, bb\}$; or $(\sqrt{L^2}) = \{aa, ab, ba, bb\}$ d'après le raisonnement à la question précédente.

   (6) est faux : avec $L = \{a\}$, on a $\sqrt{L} = \emptyset$ donc son carré aussi alors que $L^2 = \{aa\}$ donc $\sqrt{L^2} = \{a\}$.

Exercice 15

1. a) Non rationnel d'après le lemme de l'étoile.

   b) Rationnel car fini.

   c) Non rationnel : s'il l'était, son complémentaire $\{a^n b^n \mid n \in \mathbb{N}\}$ le serait et ce n'est pas le cas par le lemme de l'étoile (applique ce lemme directement au langage fourni est plus pénible).

   d) Rationnel : il est égal à $\{a^n b^m \mid n \text{ et } m \text{ sont pairs}\} \cup \{a^n b^m \mid n \text{ et } m \text{ sont impairs}\}$ qui est dénoté par $(a^2)^\star (b^2)^\star + a(a^2)^\star b(b^2)^\star$. Ou : construire un automate fini pour L_4 et invoquer Kleene.

   e) Non rationnel, allez on le fait. Si L_5 était rationnel, il serait reconnu par un automate fini à $N \geq 2$ (évidemment) états. Comme $\mathbb{P}$ est infini, il existe p premier tel que $p \geq N$ et par définition $a^p \in L_5$. Le lemme de l'étoile assure alors qu'il existe $n_1, n_2 \in \mathbb{N}$ tels que

   - $a^p = a^{n_1} a^{n_2} a^{N - n_1 - n_2}$.
   - $n_2 \neq 0$.

- Pour tout $k \in \mathbb{N}$, $a^{N+n_2(k-1)} \in L_5$.

Mais pour $k = N+1$ on aurait alors $a^{N(n_2+1)} \in L_5$, sauf que N et n_2+1 sont plus grands que 2. On en déduit que $N(n_2+1)$ n'est pas premier ce qui est une contradiction.

Exercice 16

1. *aabb* et *ababb* conviennent car ce ne sont pas des carrés.
2. Si $m \in L^c$ alors il existe $u \in \Sigma^\star$ tel que $m = u^2$ et donc $|m| = 2|u|$ est paire. Par contraposition on a donc que tout mot de longueur impaire est dans L.

   Soit m de longueur paire. Si $m \in L^c$ alors $m = u^2$ et donc pour tout $i \in [\![1, n]\!]$ on a $m_i = u_i = m_{i+n}$. Réciproquement, si pour tout $i \in [\![1, n]\!]$ on a $m_i = m_{n+i}$, m est évidemment un carré et l'équivalence de la deuxième partie de la question est démontrée.
3. $L(G, A) = \{uav \,|\, |u| = |v|\}$ et $L(G, B) = \{ubv \,|\, |u| = |v|\}$. On montre ces égalités par double inclusion : $L(G, A) \subset \{uav \,|\, |u| = |v|\}$ se fait par récurrence sur la longueur des dérivations et $\{uav \,|\, |u| = |v|\} \subset L(G, A)$ par récurrence sur la longueur des mots du premier langage.
4. Si m est de longueur impaire, il a une lettre centrale valant soit a soit b donc appartient à $L(G, A) \cup L(G, B)$. Comme $S \to A \,|\, B$, on en déduit que $m \in L(G)$.
5. Si $m = m_1...m_{2n}$ est de longueur paire, d'après la question 2 il existe $i \in [\![1, n]\!]$ tel que $m_i \neq m_{i+n}$. Supposons que $m_i = a$ et $m_{i+n} = b$ et découpons m comme suit :

   $$m = m_1...m_i...m_{2i-1}\ m_{2i}...m_{i+n}...m_{2n}$$

   Le facteur $m_1...m_{2i-1} \in L$ est un mot de longueur impaire dont la lettre centrale vaut a donc est dans $L(G, A)$. Le facteur $m_{2i}...m_{2n}$ est un mot de longueur impaire dont la lettre centrale est b donc est dans $L(G, B)$. Or, $S \to AB$ donc $m \in L(G)$. Le raisonnement est similaire lorsque $m_i = b$ et $m_{i+n} = a$: dans ce cas, la première règle utilisée pour dériver m est $S \to BA$.
6. Les questions 4 et 5 montrent que $L \subset L(G)$. Soit donc $m \in L(G)$:

   - Si $|m|$ est impaire, alors $m \in L$ d'après la question 2.
   - Si $|m|$ est paire et dans $L(G)$, il existe une dérivation $S \Rightarrow^\star m$. Parmi les quatre règles possibles au début de cette dérivation, on peut en écarter deux car A et B ne peuvent engendrer que des mots de longueur impaire (donc pas m). On en déduit que notre dérivation de m est forcément de la forme $S \Rightarrow AB \Rightarrow^\star m$ ou $S \Rightarrow BA \Rightarrow^\star m$.

     Si on est dans le premier cas alors il existe $u \in L(G, A)$ et $v \in L(G, B)$ tels que $m = uv = u'au''v'bv''$ avec $|u'| = |u''|$ et $|v'| = |v''|$. Mézalors il existe $i \in [\![1, n]\!]$ tel que $a = m_i = m_{i+n} = b$ et la question 2 assure que $m \in L$. Le second cas est similaire.

   Finalement $L(G) \subset L$ et on a l'inclusion qui manquait pour conclure.

Exercice 17

On propose pour le deuxième joueur la stratégie suivante : quel que soit le sommet s choisi par le joueur 1, choisir le sommet t tel que (s, t) est dans le couplage parfait C de G.

On peut montrer par récurrence sur le nombre de coups joués par le joueur 2 qu'une telle action est toujours possible. Le principe est que le joueur 1 commence par choisir un sommet qui est nécessairement apparié à un autre dans C, ce qui donne au joueur 2 la possibilité de suivre une arête dans C ; et dans ces conditions, le joueur 1 ne peut lui jamais suivre une arête dans C par définition d'un couplage.

Le graphe étant fini, l'un des deux joueurs finira par ne plus avoir de coup valide et d'après ce qui précède, quoi que fasse le joueur 1, le joueur 2 a toujours un coup valide : c'est donc le joueur 1 qui perd nécessairement et la stratégie proposée est gagnante pour 2.

Exercice 18

1. RAS.
2. Une stratégie gagnante pour le joueur qui commence est de commencer par prendre deux bâtons :
   - Si joueur 2 prend un bâton, il en reste 4 et joueur 1 n'a plus qu'à en prendre 3 pour gagner.
   - Si joueur 2 prend deux bâtons, il en reste 3 et joueur 1 en prend 2 pour gagner.
   - Si joueur 2 prend trois bâtons, il en reste 2 et joueur 1 en prend 1 pour gagner.

   Dans tous les cas, joueur 1 gagne.
3. Montrons par récurrence sur $n \in \mathbb{N}$ qu'une configuration à $4n+1$ bâtons est perdante. C'est clair pour $n = 0$ car dans ce cas il n'y a qu'un bâton. Soit $n \in \mathbb{N}$ pour lequel le résultat est acquis et considérons une configuration à $4n+5$ bâtons dans laquelle J doit joueur. Alors :
   - Si J prend 3 bâtons, son adversaire peut en prendre 1 ensuite et J se retrouve dans une configuration à $4n+1$ bâtons qu'on sait perdante pour J par hypothèse de récurrence.
   - Si J prend 2 bâtons (resp. 1), son adversaire peut en prendre 2 (resp. 3) pour à nouveau aboutir à une configuration à $4n+1$ bâtons, donc perdante pour J.

   Si $n = 20$ il vaut mieux commencer en prenant 3 bâtons car ceci place le second joueur dans une configuration perdante. Si $n = 21$ il vaut mieux jouer second car cette configuration est perdante.

Exercice 19

1. Tel quel, non, car ID3 s'applique à des données dont les attributs sont discrets.
2. On peut tenter de discrétiser $[0,1]^2$ en le découpant en n^2 blocs (on divise $[0,1]$ en intervalles de $1/n$ de long). Le problème étant qu'il faut potentiellement un n grand pour bien représenter les données ce qui engendre des arbres de décision d'arité très élevée.
3. Dans ce cas, on peut préférer classer les données en introduisant des seuils selon l'une ou l'autre des coordonnées.
4. Pour les données de la figure 1, notre méthode semble adaptée car les ronds et les croix du jeu d'apprentissage sont linéairement séparables via la droite $x = 0.5$. On obtiendrait donc un arbre de décision dont la racine serait étiquetée par la question "Est-ce que $x \geq 0.5$?" et dont les fils seraient étiquetés par "croix" si la réponse est non et "rond" si la réponse est oui.

   Pour le jeu d'apprentissage de la figure 2, ce n'est pas satisfaisant car on aimerait en fait pouvoir utiliser deux seuils différents selon x. Or a priori, ID3 n'autorise pas à réutiliser un attribut.
5. Il suffit de s'autoriser dans l'arbre de décision à réutiliser un attribut sur lequel on a déjà fait une séparation : de cette façon on peut créer plusieurs seuils selon une même coordonnée. A priori, cette façon de faire termine : au pire, on "encadrera" chaque point tout seul dans sa classe par des morceaux de droite (mais bien sûr ce n'est pas très pertinent de faire ça, le surapprentissage guette !).
6. On obtiendra une ligne brisée avec beaucoup de brisures entre les deux nuages de points. D'où un arbre de décision relativement complexe alors qu'à l'oeil les points semblent facilement linéairement séparables (juste, pas avec une droite verticale ou horizontale). Autrement dit, notre méthode produit un modèle complexe alors qu'un modèle très simple existe.

7. Il suffit, avant d'appliquer la méthode de la question 5, de transformer les attributs (x, y) des données en leur faisant subir une rotation à 45 degrés !

8. Dans ce cas, on peut transformer le couple (x, y) correspondant à des coordonnées cartésiennes en le couple (r, θ) donnant les coordonnées polaires du même point. Ce faisant, on rend les deux "cercles de points" linéairement séparables.

Exercice 20

1. On ne peut pas avoir $x = 0$. En effet, l'un des deux fils (au moins) incrémentera x et ce dernier n'est jamais décrémenté.

   On peut avoir $x = 2$ avec la trace d'exécution suivante : fil 1 lit $x = 0$, fil 2 lit $x = 0$, fil 1 stocke localement $v_1 = x = 0$, fil 1 écrase le contenu de x avec $v_1 + 1 = 1$, fil 2 stocke localement $v_2 = x = 1$, fil 2 écrase le contenu de x avec $v_2 + 1 = 2$.

   On peut avoir $(x, y) = (1, 3)$ avec la trace d'exécution suivante : fil 1 lit $x = 0$, fil 2 lit $x = 0$, fil 1 stocke $v_1 = x = 0$, fil 2 stocke $v_2 = x = 0$, fil 1 écrase x avec $v_1 + 1 = 1$, fil 2 écrase x avec $v_2 + 1 = 1$, fil 1 stocke $w_1 = y = 0$, fil 1 écrase y avec $w_1 + 1 = 1$, fil 2 stocke $w_2 = y = 1$, fil 2 écrase y avec $w_2 + 2 = 3$.

2. Deux fils exécutant lock peuvent générer l'entrelacement suivant : le fil 1 teste la valeur de locked et constate qu'elle vaut false donc entre dans le corps de la boucle, le fil 2 fait la même chose, le fil 1 écrit true dans locked et le fil 2 fait de même. Les deux fils ont verrouillé le verrou et se trouvent en même temps en section critique donc l'exclusion mutuelle est violée.

3. Oui, l'atomicité de test_and_set permet de garantir l'exclusion mutuelle si on modifie lock ainsi :

```
void lock()
{
    while (test_and_set(&locked)) {/* wait */}
}
```

Exercice 21

1. Un certificat dans ces conditions est un sous ensemble E des sommets de G (de taille polynomiale en $|G|$) et une vérification consiste à vérifier que $|E| \geq k$ et pour toute arête de G vérifier si ses deux extrémités sont dans E ou pas. Le tout se fait bien polynomialement en la taille de l'entrée.

2. RAS. Attention, il y a peut y avoir plusieurs sommets différents étiquetés par un même littéral.

3. Si G possède un stable σ de taille m alors les sommets de ce stable vérifient par construction :

   (1) Il y a exactement un sommet de σ choisi dans l'ensemble des (au plus 3) sommets correspondant à une même clause car les sommets correspondant aux littéraux d'une clause sont adjacents et qu'il y a m clauses.

   (2) Pour toute variable propositionnelle x, il n'y a jamais à la fois un sommet étiqueté par x et un sommet étiqueté par $\neg x$ dans σ puisque les sommets correspondants sont liés.

   Notons v la valuation qui a chaque littéral l associe 1 si l étiquette un sommet de σ et 0 sinon. Elle est bien définie d'après le point (2). Elle satisfait chacune des clauses d'après le point (2). Donc φ est satisfiable.

   La réciproque est vraie : si φ est satisfiable, il existe dans chacune des clause un littéral l_i tel que $\varphi(l_i) = 1$ et il suffit de considérer l'ensemble des sommets correspondant aux l_i dans G_φ pour exhiber un stable de taille m.

4. Construire G_φ se fait en temps polynomial en $|\varphi|$ et on vient de montrer que φ est une instance positive de 3SAT si et seulement si G_φ est une instance positive de STABLE. On en déduit que 3SAT $\leq$ STABLE et comme 3SAT est NP-dur, il en va de même pour STABLE. Comme ce problème est aussi dans NP (question 1), il est NP-complet.

5. CLIQUE est bien dans NP. De plus, si $G = (S, A)$ est une instance de STABLE, alors $G' = (S, S^2 \setminus A)$ est une instance de CLIQUE constructible en temps polynomial en $|G|$ et G est une instance positive de STABLE si et seulement si G' est une instance positive de CLIQUE. On vient de réduire STABLE à CLIQUE et comme le premier est NP-dur, le second aussi.

Exercice 22

1. 01101101 est normalisé puisque son exposant décalé vaut 110 (donc ni 000 ni 111). L'exposant décalé vaut donc $e = 6$; le décalage vaut $d = 2^{3-1} - 1 = 3$ donc l'exposant vaut $6 - 3 = 3$. La partie fractionnaire vaut $1 + 1/2 + 1/4 + 1/16$ et finalement ce flottant représente

$$(-1)^0 \times 2^3 \times (1 + 1/2 + 1/4 + 1/16) = 14.5$$

   00001000 en revanche n'est pas normalisé et il représente $(-1)^0 \times 1/2 \times 2^{1-d} = 0.125$.

2. Dans les deux cas, on obtient un résultat différent selon qu'on deamnde à une mathématicienne ou un ordinateur :

   - Dans le premier cas, l'ordinateur donnerait une valeur finie car pour n suffisamment grand $1/n$ serait suffisamment proche de 0 pour être confondu avec. Pour une machine cette sonmme est donc finie alors que $\lim \sum_{k=1}^{n} \frac{1}{k} \to +\infty$ lorsque $n \to +\infty$.
   - Dans le second cas, on devrait avoir 1 comme résultat mais une machine renverra a priori 0 car 10^{50} est très grand et donc lui ajouter 1 produit un réel dont la représentation risque fort d'être la même que celle de 10^{50}.

Exercice 23

1. Oui. D'ailleurs n'importe quelle permutation des sommets dans un graphe complet donne un cycle hamiltonien dans ce graphe puisque toutes les arêtes sont disponibles.

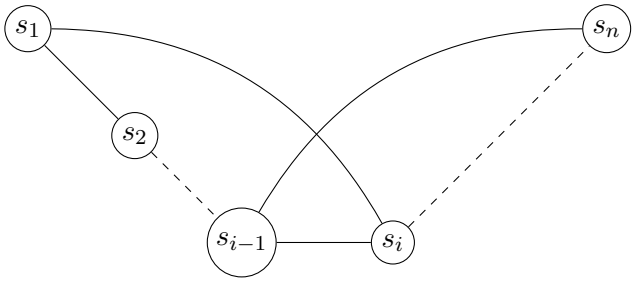
2. Il suffit de rajouter une à une des arêtes à G jusqu'à le rendre hamiltonien et il le deviendra forcément à un moment puisque qu'un graphe complet est hamiltonien d'après la question 1.

3. Soit (u, v) l'arête qu'il suffit de rajouter à H pour le rendre hamiltonien. On obtient alors un graphe qui contient un cycle hamiltonien et donc dans H il y a une chaîne hamiltonienne entre u et v.

4. Montrons par l'absurde que $I \cap J \neq \emptyset$. Dans ces conditions, on aurait $|I \cup J| = |I| + |J| = \deg(s_1) + \deg(s_n)$. Mais s_1 et s_n ne sont pas voisins dans H (sinon H serait hamiltonien et ce n'est pas) donc a fortiori ils ne sont pas voisins dans G et on en conclut que $|I \cup J| = \deg(s_1) + \deg(s_n) \geq n$. Mais d'autre part $I \cup J \subset [\![2, n]\!]$ donc $|I \cup J| \leq n - 1$. C'est une contradiction.

   On en déduit qu'il existe un élément commun à I et J et cet élément commun ne peut pas être n car ce dernier n'appartient pas à I ; s_1 et s_n n'étant pas voisins.

5. La situation dans H est la suivante d'après la question 4 (la chaîne contenant tous les s_i est hamiltonienne) :



On en déduit que $s_1, ..., s_{i-1}, s_n, ..., s_i, s_1$ est un cycle hamiltonien dans H mais ce graphe était censé ne pas en contenir : c'est une contradiction et on en déduit que l'hypothèse initiale sur G est fausse : G est nécessairement hamiltonien. On vient de démontrer le théorème de Ore.

Exercice 24

1. On peut proposer un schéma à trois entités (`eleve`, `classe`, et `prof`) et deux relations (`appartient` et `enseigne`) : la première lie `eleve` et `classe` (cardinalité 1:1 côté `eleve` et 1:$\star$ côté `classe` car un élève appartient à exactement une classe mais une classe peut contenir plusieur élèves) et la deuxième entre `prof` et `classe` (cardinalité 1:$\star$ des deux côtés car plusieurs professeurs peuvent enseigner dans une même classe et un même professeur peut enseigner dans plusieurs classes).

2. Il faut juste penser à introduire une table intermédiaire `enseignement` pour rendre compte de la relation $\star$:$\star$ entre `classe` et `prof`. On propose donc :

   - `eleves(numero_secu, nom, prenom, classe)`.
   - `classes(numero, effectif, prof_principal)`.
   - `prof(numen, matiere, nom, prenom)`.
   - `enseignement(id_prof, id_classe)`.

   L'attribut `classe` dans la table `eleves` est une clé étrangère référant à `numero` dans la tables `classes`. L'attribut `prof_principal` est une clé étrangère référant à `numen`. L'attribut `id_prof` dans la table `enseignement` réfère à `numen` dans la table `profs` et `id_classe` réfère à `numero` dans la table `classes`.

3. a)
```
SELECT numero FROM classes WHERE effectif > 35;
```

   b)
```
SELECT nom, prenom FROM profs
WHERE matiere = 'russe' OR matiere = 'latin' ORDER BY numen DESC LIMIT 3;
```

   c)
```
SELECT nom FROM profs UNION SELECT nom FROM eleves;
```

   d)
```
SELECT p.nom FROM profs p
JOIN classes c ON p.numen = c.prof_principal
JOIN eleves e ON e.classe = c.numero
WHERE e.nom = 'Béranger' AND e.prenom = "Juliette';
```

   e)
```
SELECT SUM(effectif) FROM classes;
```

   f)
```
SELECT COUNT(DISTINCT prof_principal) FROM classes;
```

   g)
```
SELECT numen, COUNT(*) FROM prof
JOIN enseignement ON numen = id_prof
GROUP BY id_prof;
```

   h)
```
SELECT numero FROM classes WHERE effectif = (SELECT MAX(effectif) FROM classes);
```