Corrigé du DS 1

① Longueur discriminante

1) Si $\mathcal{L}(\mathcal{A})$ contient un mot de longueur $\leqslant n-1$ alors il est bien sûr non vide.

Si $\mathcal{L}(\mathcal{A}) \neq \emptyset$ alors il existe un chemin dans $\mathcal{A}$ que l'on peut supposer élémentaire et donc de longueur au plus $n-1$ qui relie son état initial à un de ses états finals. Le mot qui étiquette ce chemin est de longueur au plus $n-1$ et appartient à $\mathcal{L}(\mathcal{A})$.

2) Si $\mathcal{A} = (Q, q_0, F, \delta)$ alors $\mathcal{A}' = (Q, q_0, Q \setminus F, \delta)$ convient car

$$m \in \mathcal{L}(\mathcal{A}') \text{ ssi } \delta^*(q_0, m) \in Q \setminus F$$
$$\text{ssi } m \notin \mathcal{L}(\mathcal{A})$$

3) c'est l'automate produit → cf cours.

$\mathcal{A}' = (Q', q_0', F', \delta')$ et $\mathcal{A} = (Q, q_0, F, \delta)$

On pose $\tilde{\mathcal{A}} = (Q \times Q', (q_0, q_0'), F \times F', \tilde{\delta})$

avec $\forall q \in Q, q' \in Q', \alpha \in \Sigma,$

$$\tilde{\delta}((q, q'), \alpha) = (\delta(q, \alpha), \delta'(q', \alpha))$$

On montre par rec sur $|m|$ que :

$$\forall m \in \Sigma^*, \quad \tilde{\delta}^*((q, q'), m) = (\delta^*(q, m), \delta'^*(q', m))$$

et ainsi $m \in \mathcal{L}(\tilde{\mathcal{A}})$ ssi $\tilde{\delta}^*((q_0, q_0'), m) \in F \times F'$

ssi $\delta^*(q_0, m) \in F$ et $\delta'^*(q'_0, m) \in F'$

ssi $m \in \mathcal{L}(\mathcal{A})$ et $m \in \mathcal{L}(\mathcal{A}')$

ssi $m \in \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{A}')$.

4) Si $\mathcal{L}(\mathcal{A}) \neq \mathcal{L}(\mathcal{A}')$ alors

$E_1 = \mathcal{L}(\mathcal{A}) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}')) \neq \emptyset$ ou

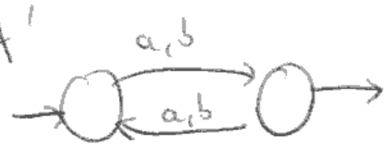
$E_2 = \mathcal{L}(\mathcal{A}') \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A})) \neq \emptyset$

La longueur discriminante de $\mathcal{A}$ et $\mathcal{A}'$ est la longueur d'un plus petit mot de l'ensemble $E_1 \cup E_2$.

Si $E_1 \neq \emptyset$ alors il existe un automate à n' états qui reconnaît $\Sigma^* \setminus \mathcal{L}(\mathcal{A}')$ d'après 2 et un à $n \times n'$ qui reconnaît E_1 d'après 4.
La q° 1 garantit alors l'existence de $u \in E_1$ t.q. $|u| \leq n \times n' - 1$.
Si $E_1 = \emptyset$ alors $E_2 \neq \emptyset$ et le raisonnement est analogue.

5) A

$\mathcal{A}'$



$\mathcal{L}(A) \neq \mathcal{L}(\mathcal{A}')$

Si on se restreint aux mots de longueur < 3 càd ≤ 2 alors les deux automates reconnaissent le m̂ ensemble $\{a\}$.

6) par rec sur $|m| \geq 1$.

si $|m| = 1$, c'est la def de φ_a :
$\varphi_a(e_i) = e_j$ ssi $(i, a, j) \in T$.

si $m = u.a \in \Sigma^{n+1}$ (on suppose le résultat valable pour ~~tout~~ $n \in \mathbb{N}^*$)

alors

$\varphi_m(e_i) = e_j$ ssi $\varphi_a \circ \varphi_u(e_i) = e_j$

ssi $\exists k$ t.q $\varphi_u(e_i) = e_k$ ($\varphi_u(e_i) \neq 0$ sinon $\varphi_m(e_i) = 0$)

et $\varphi_a(e_k) = e_j$

ssi $\exists k \mid (k, a, j) \in T$ et il existe un calcul de i à k étiquetté par u par HR

ssi il existe un calcul de i à j étiquetté par $m = ua$

7) $\varphi_m(e_1).z = 1$ ou 0 car $\varphi_m(e_1) = e_p$ ou 0

et $e_p.z = 0$ si $p \notin F$ et 1 si $p \in F$.

<u>ccl:</u> Si $\varphi_m(e_1).z = 0$ alors m n'est pas reconnu par l'automate (état d'arrivée $\notin F$ ou blocage) et si $\varphi_m(e_1).z = 1$ alors m est accepté par l'automate.

8) $\phi_m(E)$ vaut $(e_p, -e_q')$ où p est l'extrémité du calcul de $\mathcal{A}$ sur m depuis 1 et q celle du calcule de $\mathcal{A}'$ sur m depuis 1 donc

$\phi_m(E).Z = \varphi_m(e_1).z - \varphi_m'(e_1').z'$

$\varphi_m(e_1).z - \varphi_m'(e_1').z'$: 1 si $m \in \mathcal{L}(\mathcal{A})$ et $\notin \mathcal{L}(\mathcal{A}')$; 0 si $m \in \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{A}')$ ou $m \notin \mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{A}')$; -1 si $m \in \mathcal{L}(\mathcal{A}')$ et $m \notin \mathcal{L}(\mathcal{A})$

9) La famille qui engendre V_k est incluse dans celle qui engendre V_{k+1} donc $V_k \subset V_{k+1}$

10) Si $m = m_1 \cdots m_{k-1} \underset{a}{\underset{\shortparallel}{m_k}} = \mu a$

alors $\varphi_m = \varphi_a \circ \varphi_{m_{k-1}} \circ \cdots \varphi_{m_1} = \varphi_a \circ \varphi_\mu$

Ainsi, pour $u \in \mathbb{R}^n$ et $u' \in \mathbb{R}^{n'}$,

$$\begin{aligned}
\Phi_{\mu a}((u,u')) &= (\varphi_{\mu a}(u), \varphi'_{\mu a}(u')) \\
&= (\varphi_a \circ \varphi_\mu(u), \varphi'_a \circ \varphi'_\mu(u')) \\
&= \Phi_a(\varphi_\mu(u), \varphi'_\mu(u')) \\
&= \Phi_a \circ \Phi_\mu((u,u'))
\end{aligned}$$

on a bien $\Phi_{\mu a} = \Phi_a \circ \Phi_\mu$.

11) Soit $\omega \in V_k$ et $a \in \Sigma$.

ω est une combinaison linéaire des $\Phi_m(E)$ avec $|m| \leq k$ donc $\Phi_a(\omega)$ est une combinaison linéaire des $\Phi_a(\Phi_m(E)) = \Phi_{ma}(E)$ avec $|ma| \leq k+1$,

Ainsi, $\Phi_a(\omega)$ est une combi linéaire de $\Phi_{m'}(E)$ avec $|m'| \leq k+1$ donc $\Phi_a(\omega) \in V_{k+1}$

12) on sait déjà que $V_{k+1} \subset V_{k+2}$ (q°9).

Soit

$m \in V_{k+2}$, on veut m.q $m \in V_{k+1}$

Pour cela, il suffit de montrer que chaque élément de la famille génératrice qui engendre V_{k+2} est bien dans V_{k+1}. Ceci est vrai pour tout $\phi_m(E)$ avec $|m| \leq k+1$ et il reste à le montrer pour les $\phi_m(E) = |m| = k+2$.

Soit m t.q $|m| = k+2$ alors $m = \mu a$ avec $|\mu| = k+1$ et

$$\phi_m(E) = \phi_a \circ \underbrace{\phi_\mu(E)}_{\in V_{k+1} = V_k} \qquad \text{donc } \phi_\mu(E) \in V_k$$

Ainsi $\exists\, \alpha_1, \dots, \alpha_t \in \mathbb{R}^t$ et $m_1, \dots, m_t$ avec $|m_i| \leq k \ \forall i$

t.q $\qquad \phi_\mu(E) = \sum_{i=1}^{t} \alpha_i \phi_{m_i}(E)$

et $\qquad \phi_m(E) = \sum_{i=1}^{t} \alpha_i \underbrace{\phi_a \circ \phi_{m_i}(E)}_{\in V_{k+1}}$ car $|am_i| \leq k+1$

on a bien $\underline{\phi_m(E) \in V_{k+1}}$

13) Les questions précédentes permettent d'obtenir que la suite (V_k) est strictement croissante au sens de l'inclusion puis stationnaire. $\dim(V_0) = 1$ et si on note h le plus petit indice à partir de laquelle la suite est constante alors les inclusions strictes garantissant une augmentation stricte de la dimension, on obtient que $\dim(V_h) \geqslant h+1$ et puisque $\dim(V_h) \leqslant n+n' = \dim(\mathbb{R}^n \times \mathbb{R}^{n'})$ on a bien $h \leqslant n+n'-1$.

14) $\mathcal{A}$ et $\mathcal{A}'$ sont équivalents ssi $\phi_m(\varepsilon).Z = 0 \quad \forall m \in \Sigma^*$ ce qui revient à dire que les $\phi_m(\varepsilon)$ sont tous dans l'orthogonal de Z. (de dimension $n+n'-1$). Autrement dit, $\bigcup_{k\in\mathbb{N}} V_k \subset Z^\perp \iff \mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$

La question précédente garantit que $\bigcup_{k\in\mathbb{N}} V_k = V_{n+n'-1}$ et donc s'il existe un mot m t.q $\phi_m(\varepsilon).Z \neq 0$ alors il en existe un dans $V_{n+n'-1}$ et donc on a bien l'existence d'un mot de longueur au plus $n+n'-1$ qui permetrait de se rende compte que $\mathcal{A}$ et $\mathcal{A}'$ ne sont pas équivalents. La longueur discriminante de $\mathcal{A}$ et $\mathcal{A}'$ est donc inférieure ou égale à $n+n'-1$.

② Le jeu de l'awalé

15) Alice peut jouer les cases 2, 16 ou 5 (ce st les cases nonvides de son camp).

16) case 2 :

1	0	1	0	1	2
11	10	9	8	7	6
0	0	0	1	17	5
0	1	2	3	4	5

pas de récolte

case 4 :

		↑ Récolte	↑	↑	
2	1	3	2	3	4
1	1	3	1	0	7

$\boxed{\text{Gain} = 8}$

Après récolte, on obtient donc :

2	1	0	0	0	4
1	1	3	1	0	7

case 5

1	1	2	1	2	3
0	0	2	0	16	0

pas de récolte

17) 1^e situation :

Alice sème :

0	2	2	2	2	2
0	0	0	0	0	0

Elle n'a pas le droit de récolter à cause de la règle de la famine

2e situation :

1	2	2	2	2	2
0	0	0	0	0	0

Alice peut récolter les cases 10, 9, 8, 7 et 6 et obtenir un gain de 10.

après récolte :

1	0	0	0	0	0
0	0	0	0	0	0

18)
```
jeu* initialisation (void) {
        jeu* ~~jeux~~= malloc (sizeof(jeu));
        jeux→ scorej1 = 0;
        jeux→ scorej2 = 0;
        jeux→ plateauj1 = malloc (6 * sizeof(int));
        jeux→ plateauj2 = malloc (6 * sizeof(int));
        jeux→ tour = 0;
        for (int i = 0 ; i < 6 ; i++) {
                jeux→ plateauj1 [i] = 4;
                jeux→ plateauj2 [i] = 4;
        }
        return ~~jeux~~;
}

void libere (jeu* j) {
        free (j→ plateauj1);
        free (j→ plateauj2);
        free (j);
}
```

19) joueur 1 doit jouer ⟺ tour est pair.

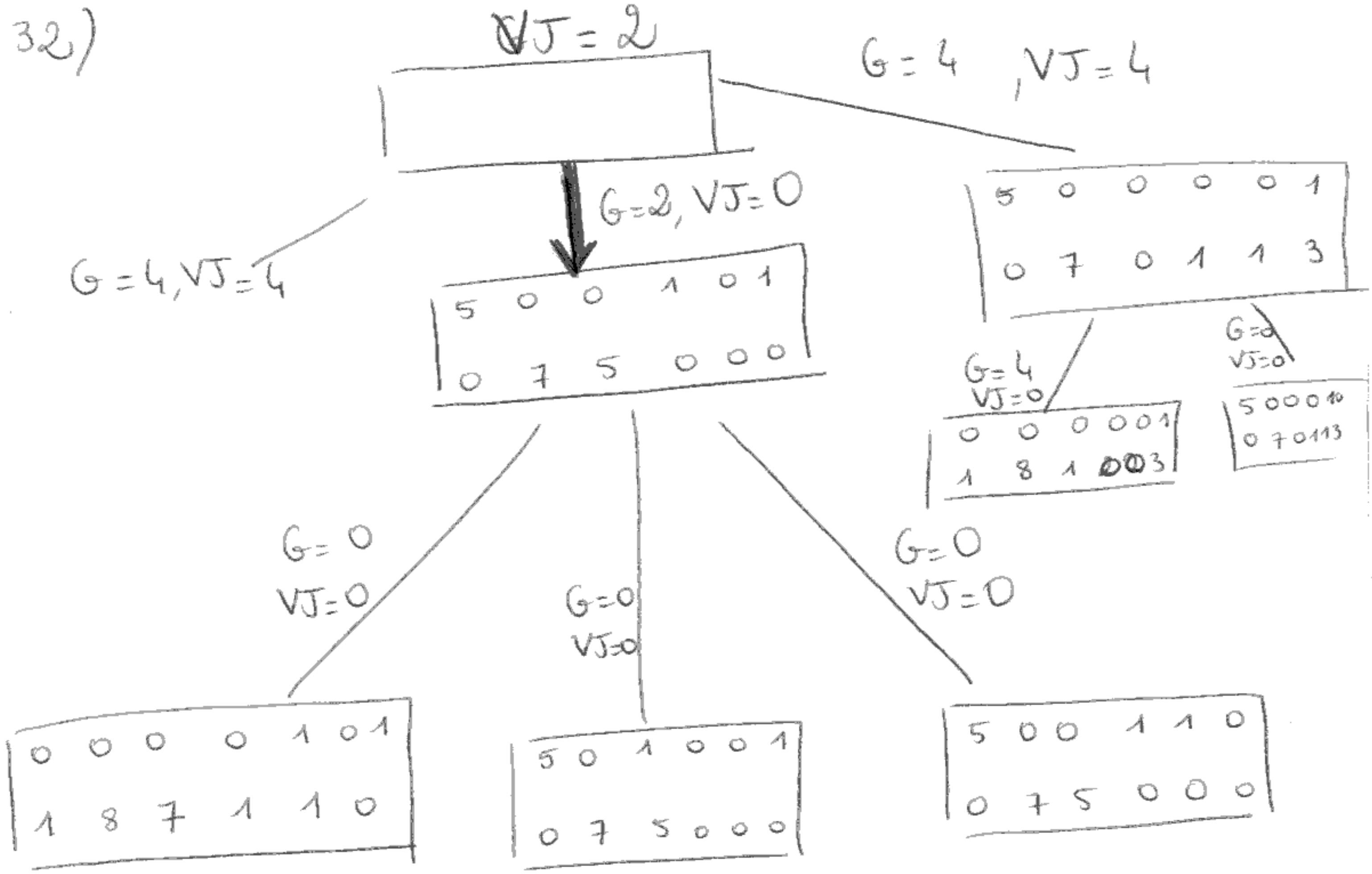
```
bool tour_joueur1 (jeu* j) {
        return (j→tour % 2 == 0);
}
```

20) Il y a 4 × 12 = 48 graines initialement mises en jeu, on ne dépassera jamais cette valeur dans une case. On a donc besoin de 6 bits

21 → 30 : cf awale.c

31, 33, 34 → cf awale.ml

32)



Selon cet arbre, Alice a intérêt à jouer la case 5.

35) Select id-Joueur from Joueur where niveau > 1900

36) Select nom, prenom from Joueur ORDER BY niveau DESC LIMIT 3

37) Select nom, prenom, count(*) as nv from Joueur join Partie on id-joueur1 = id-Joueur where resultat = 1 group by id-joueur1 having nv >= 100 order by nv DESC